Intuitive Policy Specification for Optimized Flow of Asynchronous C^3I Transmission in Operations (IPSO FACTO)

Final Report

Jim Richardson (PI)
Chris Miller
Chris Johnson
John Shackleton

Honeywell Technology Center
3660 Technology Drive
Minneapolis, MN 55418

Harry Funk

SMA Information Flow Technologies, LLC
2119 Oliver Avenue South
Minneapolis, MN 55405

Jean MacMillan
John Poirier
Kathleen Hess
Jared Freeman
Daniel Serfaty

Aptima, Inc.
600 West Cummings Park
Suite 3050
Woburn, MA 0180

Sponsored by
Defense Advanced Research Projects Agency
Contract No. DABT63-99-C-0003

May 31, 2000

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	May 2000	Final Technical Report, January 1999 to May 2000

4. TITLE AND SUBTITLE

Intuitive Policy Specification for Optimized Flow of Asynchronous C^3I Transmission in Operations (IPSO FACTO)

5. FUNDING NUMBERS

C - DABT63-99-C-0003

6. AUTHOR(S)

Jim Richardson, Chris Miller, Chris Johnson, John Shackleton (Honeywell); Harry Funk (SIFT); Jean MacMillan, John Poirier, Kathleen Hess, Jared Freeman, Daniel Serfaty (Aptima)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Honeywell Technology Center
3660 Technology Drive
Minneapolis, MN 55418

SMA Information Flow Technologies, LLC
2119 Oliver Avenue South
Minneapolis, MN 55405

Aptima, Inc.
600 West Cummings Park, Suite 3050
Woburn, MA 01801

8. PERFORMING ORGANIZATION REPORT NUMBER

None

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington, VA 22203-1714

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

IDT-R00-001

11. SUPPLEMENTAL NOTES

12A. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

We address the problem of allocating scarce communications resources in a military wide area network so that the information flows best support mission objectives. IPSO FACTO enables commanders to specify an information policy that guides automated allocation of communication resources to specific information requests. We have developed a formal mathematical model of information policy, implemented this model in software, and assessed the value of policy to resource allocation through a series of simulation experiments. We found that an explicit policy can indeed improve the total value of information conveyed in network overload conditions. However, specifying this policy is extremely time consuming. We describe a task-based, semi-automated approach to policy creation that generates policy as a side effect of mission planning. In this approach, a hierarchical mission plan is constructed from task templates; each template carries information policy elements to support the task. As the templates are arranged into a task hierarchy, the policy elements are merged to form a complete policy.

14. SUBJECT TERMS	15. NUMBER OF PAGES
resource allocation; information policy; task model; mission planning	80
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

Table of Contents

List of Figures

List of Tables

Section 1
Introduction

1.1 The Problem

The concept of "winning the information war" has always been an important principle in the history of warfare. In the past, the military often struggled with the problem of not having enough access to information. Over the years, new technology has produced the opposite problem—today there often exists such a wealth of available information that it becomes difficult to filter out the crucial data.

However, there is another fundamental problem involved in winning an information war, one of neither too much information, nor too little, but rather one of *allocation*. Namely, given a world where the resources that generate, process, and route information are limited, these resources must be allocated in the most productive means possible in order to succeed. The overhead costs associated with performing this allocation, though, are usually high compared to the benefits, especially when the meaning of "productive" changes from moment to moment in the heat of battle.

While this problem exists to some extent for most participants on the battlefield, it most affects the tactical commander, whose decisions about resource allocation most critically affect the conduct of the battle, yet whose time and decision making are already taxed to the limit. Therefore, to require the battlefield commander to create frequent, detailed policies for information allocation is to place a novel, extraordinary burden where it can be least easily assumed—and where the consequences of failing to do so will be most severe. However, the commander is *exactly* the one who needs to make these decisions. Delegating the task to subordinates, or worse, to automated planning or management assistants, invites mismatches between the commander's goals for the battle, and the information policies actually enforced. So how can a commander intuitively and successfully communicate his information needs directly to a complex network management system, such that crucial information gets to the right soldier at the right time?

1.2 A Solution

A year ago we proposed IPSO-FACTO—or Intuitive Policy Specification for Optimized Flow of Asynchronous C^3I Transmissions in Operations—as a solution to this dilemma. This report details the IPSO-FACTO approach and the progress we have made during its first year of development. The IPSO-FACTO project is part of DARPA's Agile Information Control Environment (AICE) program.

IPSO-FACTO enables commanders to specify an information allocation *policy*, which in turn informs an automated communications resource management system. In this approach, a policy represents the commander's desired allocation of communications resources during the execution of a mission; the policy takes the form of a set of general and specific statements about the priorities, constraints and objectives for information flow. However, we view policy as

applicable to more than just military communications; the concept is central to any problem in which a decision maker must precisely guide the allocation of any type of resource—in domains ranging from cockpit display layout to refinery operations. Essentially, our approach addresses the following issues related to creating and applying policy:

- Architectural concept
- Policy representation
- Policy application
- Multi-user policies
- Task-based policy construction

1.2.1 Architectural Concept

The purpose of AICE is to manage resource allocation in military communications networks according to commanders' policies, so that when resources are insufficient to carry all the requested network traffic, resources are allocated to carry the information that is most important to mission success.

Figure 1 is a simplified version of the AICE architecture, developed by the IPSO-FACTO project and our collaborators in DARPA's AICE program. AICE, taken as a whole, allocates resources in a collection of interconnected networks. The networks employ a variety of technologies such as ATM, Satcom, and wireless IP. Applications residing in host computers send information to each other across these networks. Each transfer requires a channel between the information source application and one or more information destination applications. One of the applications must first request a channel to accomplish the transfer. AICE, guided by commanders' policies, either constructs the channel or not, depending on current network resource commitments and certain attributes of the request. In granting the request, AICE may preempt a previously allocated channel that has lower importance.

The three principal AICE components are Information Policy Management (IPM), Adaptive Information Control (AIC), and MetaNet. IPSO-FACTO concepts are implemented in the IPM component. IPM accepts multiple commanders' policies regarding network resource allocation, resolves any differences among them, and computes the importance of individual requests in response to queries from the AIC component. AIC tracks current network resource usage and decides whether or not to grant the request. MetaNet interacts with the underlying networks to set up the end-to-end channel; it provides a uniform interface to heterogeneous networks.

Granting a channel request is not an all-or-nothing proposition. Rather, each request includes a set of quality of service (QoS) requirements. QoS requirements are expressed in terms of bandwidth, delay, and loss ratio. (Other QoS dimensions could be added, such as privacy.) For each request that it grants, AIC assigns a QoS value in each dimension. For instance, the request could include a 1 Mbit/sec bandwidth requirement, but AIC may assign only .5 Mbit/sec due to network loading. The ability to allocate varying QoS levels gives AIC additional flexibility in maximizing the value of information transferred across the networks.

A more detailed treatment of the architecture is available in the AICE Interface Requirements Document, produced by TRW.

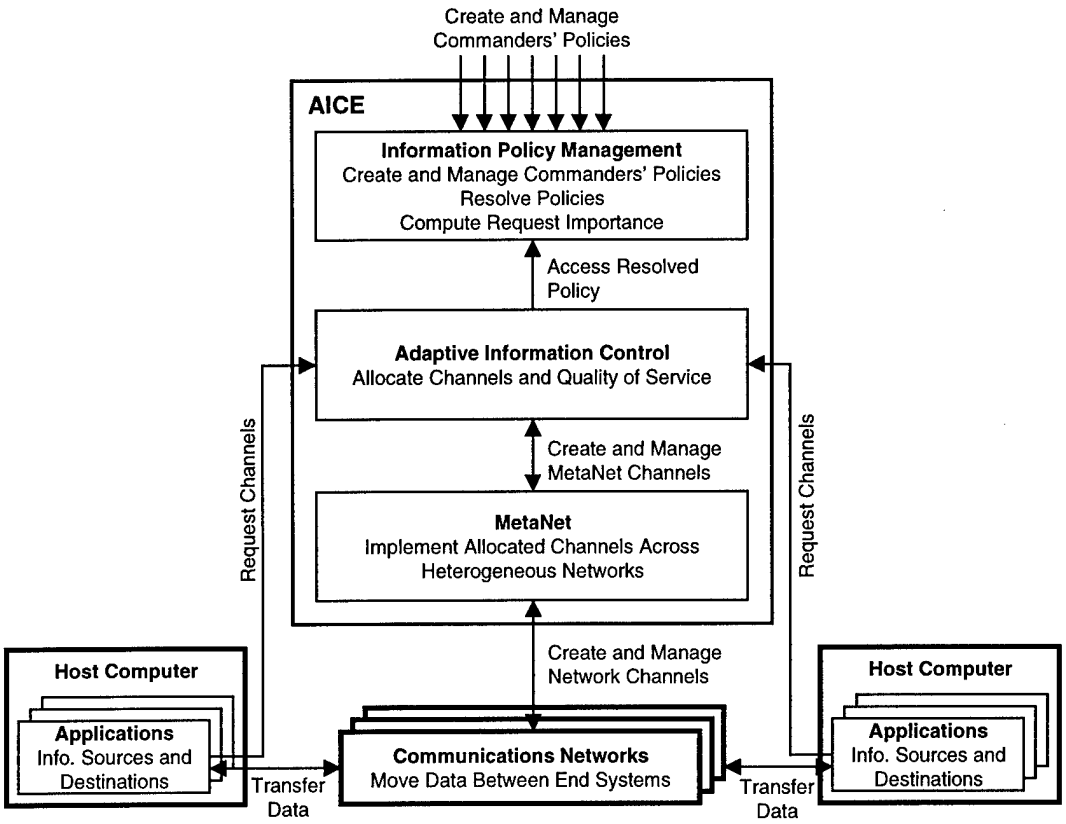

Figure 1. Simplified AICE Architecture

1.2.2 Policy Representation

We have developed a simple, expressive method of stating information policies—that is, statements about the value of requested information. Each commander's policy is composed of a set of *policy elements* that assign importance to information requests based on the following attributes:[1]

- **Owner**: who made the request?
- **Source**: where is the information coming from?
- **Destination**: where is the information going to?
- **Content**: what kind of information is it?

This representation is illustrated more conceptually in Figure 2. Each policy element can be seen as defining a region in the multi-dimensional space of a policy. A region may be based on a single dimension (e.g., requests for weather forecasts `[content]` get an importance of 0.2) or on a combination of dimensions (e.g., requests supporting the *Zone Reconnaissance* task `[owner]` for weather forecasts `[content]` from Satellite 476B `[source]` to 3rd Air Calvary Division `[destination]` get an importance of 0.8). Importance is shown here as a single numerical value, but in a more general form is a shaped function with different profiles which

[1] These parameters are tailored to the domain of information resource management; policies for other kinds of resources would depend on different characteristics.

indicate the importance of matching a requestor's stated need along any of the policy's represented dimensions. In our current representation, policy elements are sequenced—generally from most to least specific—to indicate their order of precedence.

	Owner	Source	Destination	Content	Importance	
(	{W1}	{S1}	{D1}	{C1}	.9	)
(	{W2}	{S2}	{D2}	{C2}	.8	)
(	{W3}	{S3}	{D3}	{C3}	.5	)
(	*	*	*	*	.1	)

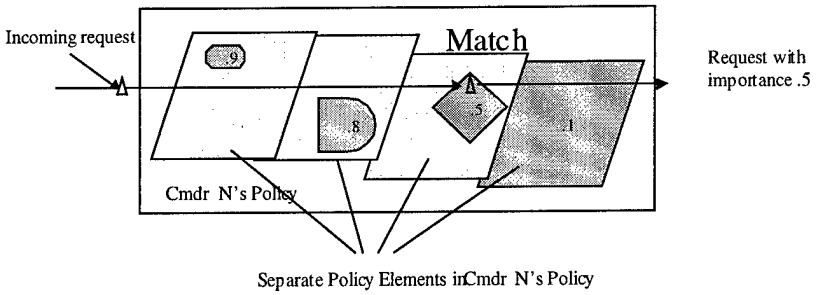

Figure 2. Representation of Policy in IPSO-FACTO

1.2.3 Policy Application to Resource Management Plans

Given the policy representation presented above, how is the importance of an incoming request actually determined? Each request is associated with a particular commander's policy. As shown conceptually in Figure 2, the incoming request proceeds through that particular policy space—through the sequenced set of policy elements issued by the commander—and is assigned the importance of the first region it hits (i.e., the first policy element it matches). At present, matches must either be exact or based on "descendant-of" relationships, but future implementations will allow partial and "best" matches. This may well obviate the need for sequencing.

1.2.4 Cross-Echelon Policy Construction

To this point, we've discussed policy as if a single commander makes all the decisions about resource usage. In realistic military operations, this never happens; multiple commanders at multiple levels control—and sometimes even compete for—the same pool of resources. At the very least, each commander must allocate resources in accordance with the policies of those above. This same command structure is preserved in our concept of policy capture and application. As illustrated in Figure 3, policies exist at nodes in a command hierarchy. As each request comes in, it is matched against the commander's policy it explicitly supports, but then it must also be matched against the policy of the superior commander—and so on, up the chain of command.

Commanders control how the importance produced by their policy should be resolved with the importance produced by their immediate subordinate's policy. For instance, a commander can currently choose from the following resolution methods:

- **Maximum**: "The request can be no more important than what I say."

- **Minimum**: "The request must be at least as important as what I say."
- **Linear Combination**
  - **Combine(1)**: "Ignore my subordinates, I'll decide how important this request is."
  - **Combine(0)**: "Ignore me, let my subordinates decide how important this request is."
  - **Combine(x)**: "Compromise and make the request's importance a combination of mine and my subordinate's, using a ratio of x.

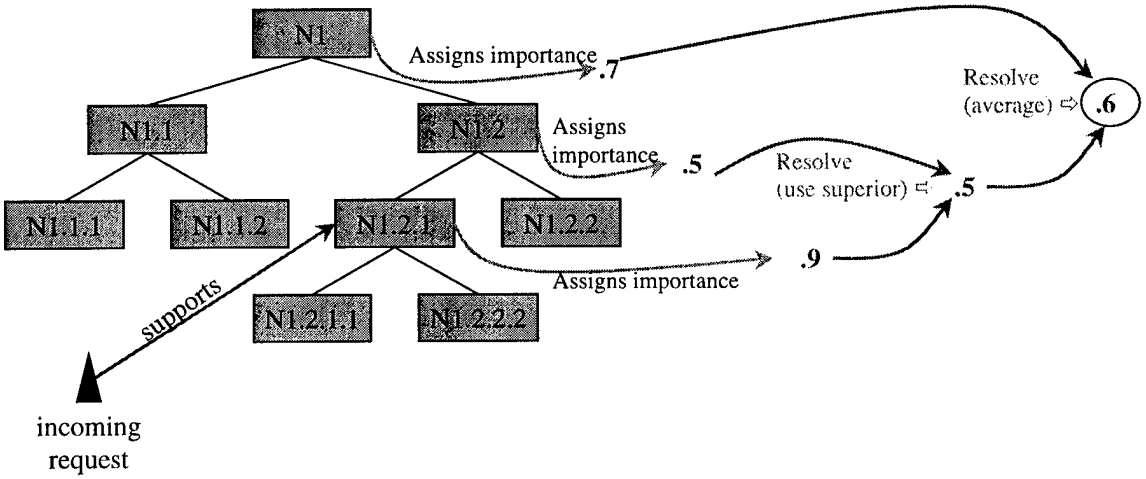

Figure 3. Cross-Echelon Policy Application and Resolution

1.2.5 Task-Based Construction of Resource Management Plans

While less onerous than making decisions about individual resource allocations, creating policy is still a lot of work—more work than a commander should have to worry about. Commanders (and their staffs) should be able to concentrate completely on planning the overall goals, strategy, and tasks of a mission; currently, a commander creates a mission plan, and then must also go back and interpret the plan again to produce an information policy.

IPSO-FACTO can reduce this extra effort by providing a tool that allows commanders to build mission plans from a library of standard and archived mission tasks and their associated information policies. As a result, commanders would be able to focus on the plan itself—just as they already do—without the distraction of also creating detailed policy about the plan's information needs, and yet still retain the benefits of doing so. Such a tool is possible because:

- Information needs are *dependent* on goals and tasks. In other words, the importance of any given type of information at any given time is based on how well that information advances the achievement of the current goal or task.
- Mission planning, as with most domains, possesses a set of common goals and tasks that cover a good majority of situations.

Therefore, our ultimate vision for the IPSO-FACTO system involves a commander piecing together a mission plan by selecting task "templates" from this library of common tasks. Each template would already include a set of information policy elements, as well as other useful knowledge, such as alternative procedures for achieving the task. After adapting each template to

information specific to the current situation and prioritizing the tasks, the commander would possess a mission plan complete with a workable, detailed information policy to guide the allocation of information resources. Of course, we would expect that the commander may want to alter aspects of the policy to better fit unique elements of the situation, but this would still require substantially less work than developing an entire set of information requirements from scratch. In this way, policy can be derived from the dependencies and priorities associated with mission tasks, automatically producing a majority of the detailed intent and allocation statements appropriate for a mission.

1.3 Current Status

We have implemented and evaluated a prototype version of IPSO-FACTO. Initial results show that the policy representation we have provided is very expressive, but, as expected, time-consuming and error prone to generate manually. Nevertheless, such a policy does enable more human control of algorithms for automated resource allocation, and when evaluated against an independent, mission-based measure of worth, can improve the performance of the allocation dramatically. Therefore, we believe that detailed policies that assign importances to resources based on semantically important characteristics are crucial to the improved allocation of these resources, and that intelligent tools are necessary to make the creation of such policies feasible.

The rest of this report discusses each of the above issues in more detail:

- **Section 2** discusses why policy is so important in domains involving resource allocation.
- **Section 3** presents the format developed and used for policy in IPSO-FACTO and the effects of applying policy in AICE, based on evaluation tests run by TRW Inc.
- **Section 4** discusses our approach to resolving and applying policy when varying policies are made by multiple decision makers at multiple levels of command, and the effects of this cross-echelon approach in AICE.
- **Section 5** presents future directions for IPSO-FACTO and the use of policy to guide resource allocation and other kinds of optimization processes.
- **Section 6** discusses the need for a task-based tool to aid policy creation, and our recommendations for such a tool.
- **Section 7** presents implementation and test considerations addressed in the year 1 prototype.

Section 2
Need for Policy in Resource Allocation Domains

2.1 The Use of Policy in Distributed Decision-Making

AICE is operating on an instance of a general class of resource allocation problems where decisions about how to allocate resources must adapt to context and situation, but were the overhead *costs* and complexity associated with deciding how to allocate resources are greater than the human portions of the system can bear. The result is suboptimal system performance for any of a number of reasons. Expert human decision-makers may not be available or may be overstressed, and thus less expert decision-makers must be relied upon. In other situations, no decision-maker at all is available and the potential for decision (and, therefore, adaptation) must be eliminated by making the system behavior static. If expert decision-makers are available, they may not have all pertinent data, or if they do, they may not have sufficient time to review it and to make well-considered decisions.

These problems will become increasingly critical in the domain of communications resource allocation in the future digital battlefield. In principle, each commander *knows* the worth or importance of every request for use of communications resources made by those under his command[2]. If he had unlimited time and complete situation awareness, he could make very good decisions about whether or not every incoming request should get access to resources or not. Of course, this is not remotely the case. Even if the commander were to do nothing but make communication resource allocation decisions, future digital battlefield scenarios for even moderately sized forces imply that communication resource allocation decisions will need to be made far faster than any human will be able to process them.

Again, in principle, the commander could delegate the task of making resource allocation decisions to subordinates, but this invites mismatches between the commander's goals and intentions and the decisions made. One way to mitigate this problem is for the commander to convey "policy" to his subordinates. This shows the role of policy as an expression of the commander's intent and goals, but since the specific decisions to be made are not yet known, it is necessarily abstract and general and must be applied to make specific decisions as they arise.

Of course, the availability (and cost) of sufficiently trained and intelligent decision-making subordinates is also doubtful—and therefore, the notion of simply allocating the set of resource allocation decisions to subordinates is also untenable. Increasingly commonly, we are turning to automation to ease the commander's decision-making burdens. Automated resource allocation

---

[2] Actually, this claim is arguable. Commanders, like all of us, regularly make bad decisions. To be fully confident of the worth of any communication request, we should probably 'rerun' the battle as many times as there are possible alternative worth decisions, holding all other variables constant, and view the impact of the various decisions on battle outcome. Even when this is possible in simulation, it is almost always impractical to do in real time when the decision is needed. Thus, a more accurate statement would be 'each commander is the best available resource for determining the worth or importance of every request for communications resources made by those under his command.'

algorithms, such as those being developed for the Agile Information Control (AIC) layer of the Agile Information Control Environment (AICE) program, can make decisions much more rapidly than any humans can, but otherwise the same problem described above, for allocation of decisions to subordinates, exists. If the automation is to make adaptive decisions which truly extend the commander's intent and satisfy his goals, they too must be informed of his 'policy' in a format that they can use-a much more difficult constraint then when the aide is human.

2.2 The Need for Flexible, Adaptable Policy

It is important to note that *all* resource allocation algorithms (and, in fact, all computational optimization approaches) must rely on an optimization function—which is exactly a statement about how to determine value or, in our terms, a 'policy'.

An innovation of the IPSO-FACTO approach described in the rest of this document is the realization that policy generally can't be stated once and for all, but will need to be modified, updated, reviewed and applied as context and objectives change. While this fact is generally accepted of human-human interactions over delegated decision-making, it is less so in human interaction with resource allocation automation. In military domains, for example, the creation and conveying of 'commander's intent' from a commander to his human subordinates is a subject of military training courses, specially formulated documents, and intensive research. It is understood that a carefully crafted intent statement, *customized to the current situation, resources and goals*, is critical to getting subordinates to behave appropriately.

By contrast, many more or less intelligent resource allocation approaches have assumed a fixed, or only coarsely modifiable value function—which corresponds to the policy by which the algorithm makes its decisions. Resource allocation algorithms, scheduling systems, even control laws assume a fixed notion of what is 'good' or to be sought after.

This is true of many current and proposed approaches to communications resource allocation, which assume that completing the maximal number of communications requests with available resources is always to be desired. Here are two additional examples from non-communications domains:

- Honeywell's current control theoretic approach to optimizing the allocation of military aviation resources to service targets in the JFACC domain has been designed to make complicated tradeoffs between probability of achieving a desired degree of target destruction with a minimal 'cost' (complexly defined) weapon system. While the number of factors considered is impressive compared to other similar systems, the fundamental assumption is that JFACC will always want to use the smallest weapon that has a 'reasonable' likelihood of achieving the desired degree of destruction. Recent experiences in Kosovo show the weakness of this assumption. There, during much of the war, JFACC was instructed to systematically use less than ideally effective munitions in order to minimize civilian casualties and collateral damage.
- Honeywell's Flight Management Systems, used on many commercial and military aircraft to compute optimal flight paths and then manage aircraft flight performance to optimize fuel consumption within the constraints imposed by air traffic control and by passenger

comfort. While this is certainly *a* desirable goal, and it is the criterion on which Honeywell sells FMSs, it is not always the way in which pilots want to make their flight routing decisions. Sometimes on-time arrival is more important than fuel savings, for example. The tradeoff space between weather avoidance and fuel consumption is a complex one in which pilots have broad latitude but are under pressure from their organizations to make efficient decisions. In spite of the fact that FMSs have been built with a complex set of pilot constraint input capabilities (so much so that the workload and training requirements for interacting with them are a problem), pilots still occasionally have 'surprises' when the FMS recommends a path that makes no sense to them, and sometimes have to resort to 'tricking' the FMS into giving them the type of flight path they know is right.

The point of these examples is not to imply that Honeywell makes bad systems. Rather, it is to illustrate that even sophisticated optimization approaches will never be right all the time if they are static and the parameters which define 'goodness' or 'value' in the domain are not static. Furthermore, when such approaches are wrong, humans have to resort to extraordinary mental (and sometimes physical) gymnastics first to realize that they are providing undesirable recommendations, and then to develop an approach that works. For example, the JFACC operations developer must realize that the sortie recommended for a target as 'optimal' is only optimal under a set of value criteria that don't quite match those that are currently his 'real' priorities—and he must go back over the algorithm's recommendations and tweak them in accordance with other factors which he knows about but the algorithm does not (e.g., the importance of minimize collateral damage, and civilian and friendly casualties). Analogously to the FMS example above, this may require unintuitive alteration of inputs to ('tricking') the algorithm to force it behave in the desired fashion, especially when the algorithm and operations developer are not using the same concepts and language.

Note that the above claim implies that there are some domains in which 'goodness' or 'value' are static enough that sophisticated and flexible policy specification capabilities are not necessary or warranted. In these cases, a static valuation approach built into the resource allocation algorithm, perhaps augmented with a few human-adjustable, 'slider bar' dimensions of value are sufficient. We claim, however, that these domains are far less frequent than the static-policy resource allocation approaches which have been built for them. More information about domains in which flexible policy specification is needed and useful will be provided below.

2.3 The Importance of Separating the Value Model from the Decision Algorithm

What is needed is a human usable approach to creating guidance statements, or 'policy'. That is, we must allow the human to specify how solutions should be valued—in a way that is in turn usable by the resource allocation algorithms.

Another innovation of the IPSO-FACTO approach is the separation of the specification of policy from the 'act' of decision-making about resource allocation. Making 'policy' a first-order, human manipulable entity *at run-time* (as opposed to design time) sets up the conditions for the degree of flexibility in policy specification that are needed to meet the demands described above. Making that same policy readable by the resource allocation algorithm, in turn, sets up the conditions to enable flexible tuning of resource allocation decisions. The resource allocation

algorithm does the same job and follows the same process, it simply does it with a different valuation function and, therefore, the results are different—and more flexibly tuned to the situation and to human intent and value.

Fortunately, many types of computational optimization approaches support or are easily adaptable to making the value model a first-order entity. (Examples are discussed below.)

In fact, we suspect that the only types of decision-making approaches which clearly do not offer the conditions described above are those in which the value criteria are built into (that is, are implicit or embedded in) the algorithm itself. Heuristic classification in rule based systems is the prime example of such an approach. Here, the order in which rules are selected and used (by either forward- or backward-chaining methodologies) embeds a very implicit notion of policy—but the ability to access, review or modify that policy is not only unavailable to the end user, it is essentially unavailable even to the designer who generally can't modify rule firing orders with predictable consequences.

Of course, even when a potential for separating resource allocation algorithms from the value model they use exists, there is no guarantee that the user will have access to the value model. As mentioned above, many resource allocation algorithms which are currently available use a static value model which, although it could in principle be modified, no hooks are provided to do so.

The alternate approaches to resource allocation described above can be depicted as in Figure 4. Figure 4A shows the state of common practice where a value model exists more or less explicitly in the resource allocation approach, but it is static. Furthermore, it is at least potentially viewable and modifiable by the human user, but this capability is not currently enabled—which is the reason that the value model remains static. Figure 4B represents the difficult case where the value model exists only in an implicit sense, embedded in the architecture of the resource allocation algorithm itself. Heuristic classification approaches (including many non-numeric rule-based systems) are examples of this class. The desired state for any system in which a more flexible and tunable value model would provide better decisions is represented by Figure 4B. Here, the user has the capability to create many alternative value models, each tuned to a different situation, goal set or mission plan, and assert them to a resource allocation function that then operates using them to provide solutions which are more in keeping with user intent.

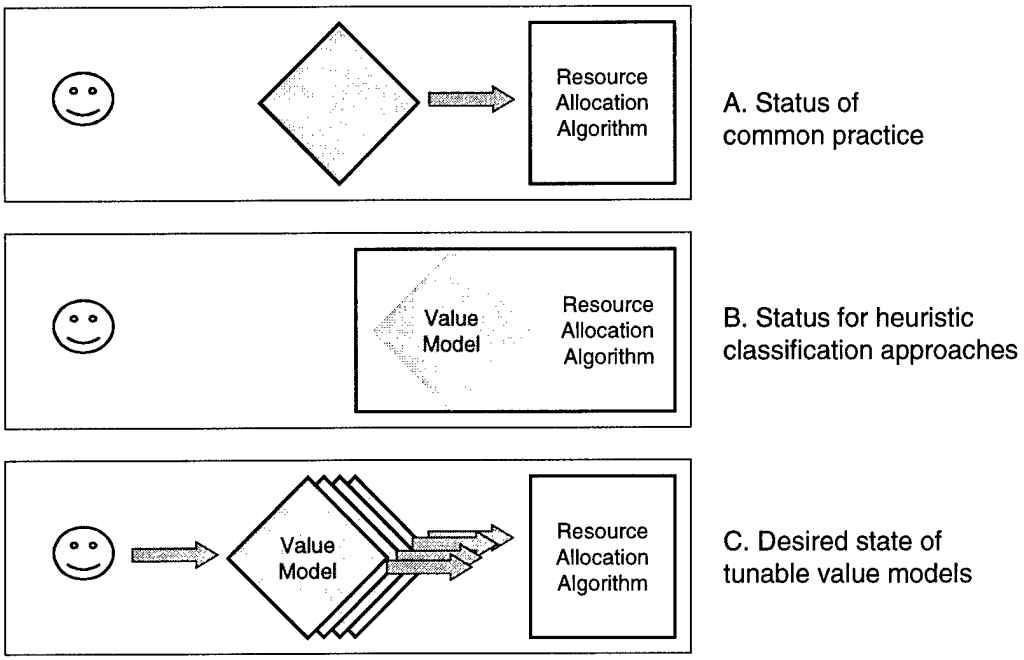

Figure 4. Three Approaches to Resource Allocation

2.4 Generalization to Other Domains

IPSO-FACTO has provided a method for creating resource allocation approaches as illustrated in Figure 4C—or of transforming a situation such as depicted in Figure 4A into that depicted in Figure 4C. IPSO-FACTO does this via a human-tunable policy specification which constructs a value model. Both the representation and the resulting model have been designed for the specific domain of military network communications—and to interact with the specific resource allocation approaches defined in the AICE program's Agile Information Control. Nevertheless, with suitable modifications to the policy representation and with some negotiation over how the value model is to be accessed or is to report values to the resource allocation algorithm, we believe that the general approach we have developed for IPSO-FACTO should be generalizable to any domain except those falling into the category depicted by Figure 4B.

Some classes of resource allocation approaches that could benefit from the IPSO-FACTO approach include descent methods, quasi-Newton methods, Lagrange multipliers, linear programming and the like.

Some comparatively straightforward applications of the IPSO-FACTO approach to other resource allocation domains might include:

- Logistics operations for military and commercial domains
- Service allocation in telecommunications
- Service allocation in medical triage
- Service allocation security protection
- Personal time schedulers
- Corporate R&D budget allocators
- Public Policy domains

- Prioritization of maintenance operations
- Prioritization of airline resources (crews, gates, takeoff & landing slots, etc.)
- ...the list seems endless.

One potential hurdle (though not an insurmountable one) to the generalization of IPSO-FACTO's approach to using policy to constructing a value model is that, to date, our vocabulary for expressing policy statements is inherently monotonic. That is, the concept of 'value' embodied in each policy statement must be expressed by a simple quantity that could be regarded as either a 'cost' or a 'benefit', but not both. In the remainder of this paragraph, lets assume we're talking about 'value' as positive—as benefit—to simplify the discussion, but with the awareness that we could as easily be talking about negative costs. Under this monotonic notion of benefit, when a request is satisfied, it earns the value assigned to it in the value model. There is no notion of 'cost' or negative benefit in this model except a sort of 'opportunity cost'-- the fact that requests exceed resources and, therefore, that the decision to satisfy one request (and earn its associated benefit) means that we won't be able to satisfy some other request (and therefore have failed to earn its benefit).

In fact, in many domains, this is a simplification. Even in IPSO-FACTO's domain of network communications, the use of resources to satisfy a request may have costs that aren't easily captured as simply a lack of benefit. For example, some communication channels may reveal information to the enemy, some are less reliable than others, and some may actually involve costs in the form of wear and tear on equipment or dollar costs to rent space on networks not owned by the military. We can embed some of these non-monotonic costs in the value structure by creating policy rules that reduce their associated benefits—though it is probable that such representations might demand the inclusion of knowledge about specific resources used (e.g., which transmission channels)—something we have not yet done for IPSO-FACTO. For example, satisfying any request for the 73rd ARTY battalion may have a benefit of .6, but satisfying a request using point to point RF transmissions will have a value of .2. The reason for this might be because RF transmissions would give away friendly positions on this covert mission, but that rationale is no where represented explicitly. In general, this approach to preserving monotonicity will likely prove inelegant, non-expressive, confusing and probably ultimately unusable if the number and variety of such cases is great.

Instead, we will need to begin exploring policy representations that both increase and decrease value in a non-monotonic fashion for any domain in which users will want to write valuation rules about both how to achieve benefit and how to avoid cost. Such domains might include several on the list above including logistics operations and medical triage.

Section 3
Formal Definition of Policy and Its Effect on Resource Allocation

In today's operations, policy guidance is conveyed from the commander to subordinates by means of something like an OPORD, a structured free text explanation of the planned operation and the associated implications. Unfortunately, as is demonstrated again and again in operations, this free text conveyance is often misstated and/or misinterpreted (to a greater or lesser degree). This occurs even though the humans involved share common training and context information. For future systems that pair a human with automation, this common basis is reduced, and the potential for misinterpretation grows.

What is needed is a more precise, mathematical formulation of policy, one in which the syntax and semantics are well defined and unambiguous. We have created such a formulation, based on the terms and operators that we felt a commander would commonly use to discuss his intent with regard to the use of communication resources.

Having defined policy precisely, the main question is: does policy improve system performance? We have supported TRW in the evaluation of Information Policy Management (IPM, see section 1.2.1). Evaluation results show that our policy representation is a reasonable first step, but there is room for improvement.

This section will treat policy as a single coherent view (which we term the "resolved policy") and associated evaluation results Section 4 will introduce the complexities associated with multi-user policies and present associated evaluation results.

3.1 Mathematical Formulation of Resolved Policy

IPM provides two different views of policy:

- A simple view of policy provided to the Adaptive Information Control (AIC) layer of AICE, This is the form that the optimization component needs, and is essentially the form that would be created if there were only a single policymaker. In fact, this form of policy is the result of combining multiple commanders' individual policies and resolving their differences, and so is called the *resolved policy view.* This view is defined here.
- A *cross-echelon policy view* provided to commanders who create and maintain policy. This view represents individual commanders' policies and the parameters governing their combination into a resolved policy. This view is defined in Section 4.

We have found the exercise of creating a mathematical formulation very useful. We have been forced to clarify our own thoughts about what a policy means. It has helped us to identify ambiguities in loosely-stated examples of "commander's intent". Finally, it has proved to be a good tool for communicating policy concepts across the AICE team.

3.1.1 Requests

A request r_k is of the form $(w_k, s_k, \mathbf{d}_k, c_k, u_k)$, where:

- w_k is the *owner*—the creator of the exchange request. The set of all possible owners is W.
- s_k is the *source*—an application entity that participates in an exchange. In a one-way exchange, the information comes from the source. The set of all possible sources is the set of applications A.
- $\mathbf{d}_k$ is the *set of destinations*—a set of application entities that participate in an exchange. In a one-way exchange, the information goes to all destinations. The set of all possible destinations is also the set of applications A.
- c_k is the *exchange characterization*—a description of the information content of the exchange. The set of all possible exchange characterizations is C.
- u_k is the *flow characterization*—a function that defines the owner's quality of service requirements and desires for the exchange. The set of all possible flow characterizations is U.

3.1.2 Resolved Policy Functions

A resolved policy provides, for each request r_k, an importance value i_k. The importance is a function of the request's owner, source, set of destinations, and exchange characterization:[3]

$$i_k = p(w_k, s_k, \mathbf{d}_k, c_k)$$

The set of possible importance values is I, and has been defined as the set of real numbers between 0 and 1. More formally, a resolved policy is a function $p : W \times A \times \mathbf{P}(A) \times C) \rightarrow I$, where $\mathbf{P}(A)$ is the powerset of A (the set of all subsets of A).

3.1.3 The Meaning of Importance

What do importance values mean? How does the assignment of a particular importance value to a request affect the fulfillment of that request? The meaning ascribed to importance by the commander and AICE must match (or at a minimum, be compatible) for the results of the optimization process to satisfy the commander's intent.

While there are many possible interpretations of importance values, any interpretation should have the following properties:

- The higher a request's importance, the more likely it will be fulfilled, and fulfilled with the quality of service that best meets the owner's requirements, subject to resource constraints. Note that it is still possible that low-importance requests are fulfilled while some high-

---

[3] The ability of policy to take a request's owner explicitly into account, as well as its source, destinations, and exchange characterization, is an extension of the formalization of policy described in [].

importance requests are not, or are fulfilled with reduced quality of service, because the high-importance requests require resources that are not available.

- An importance value of zero should not prevent the request from being fulfilled. The request should be fulfilled if possible and if doing so does not decrease the total information delivery value.

In AICE, the AIC component determines the meaning of importance by the way it uses importance values to guide resource allocation. AIC treats resource allocation as an optimization problem, and attempts to maximize some measure of *total information delivery value*. This value should estimate the contribution of the request satisfaction to the likelihood of mission success. There are various ways to estimate this information delivery value. A simple approach would be:

$$(1) \qquad value = \sum_k i_k \cdot u_k(QoS_k)$$

where i_k is the importance that IPM assigns to request r_k, QoS_k is the quality of service assigned to the request, and u_k is a flow characterization—the function that defines the utility of this quality of service to the owner.

Another measure of total information delivery value is delivered value as a function of importance. This measure encourages full satisfaction of requests with a particular importance i before allocating any resources to requests of lower importance. It models the intent of a commander who wants the most important information to get through, even if that prevents many medium-important requests from being fulfilled. Mathematically, value is not a scalar value but a function $value : I \rightarrow \mathbf{R}$:

$$(2) \qquad value(i) = \sum_{i_k = i} u_k(QoS_k)$$

AIC's optimizer can compare the values of any two resource allocations using the following order relationship on the value functions:

$$value_1 > value_2 \Leftrightarrow \exists i \in I \ (value_1(i) > value_2(i) \quad \wedge \quad \forall i' > i \ \ value_1(i') = value_2(i'))$$

In other words, $value_1 > value_2$ if the total utility delivered under the two resource allocations are equal for each importance value down to some level i, and the first resource allocation delivers greater utility at importance level i. This value measure is used by at least one of the AIC implementations.

3.1.4 Dependencies

Equations (1) and (2) have a common property: the value assigned to fulfilling one exchange request is *independent* of the value assigned to fulfilling any other request. There is no way of representing situations such as:

- There is value in fulfilling either of two requests, but no additional value in fulfilling both.
- There is value in fulfilling both of the requests, but none in fulfilling just one of them.

Dependencies such as these arise because information deliveries support mission tasks, and there are dependencies among these tasks or dependencies in the information required to support the tasks. Our cross-echelon policy representation (REF _Ref478198959 \r \h Section 4) currently provides only a skeletal representation of tasks, and does not represent dependencies among tasks or the information requests that support them. We believe that a richer representation of tasks within a policy will allow the policy to more accurately guide resource allocation and at the same time be just as easy to construct. This is an area for future research; see Section 6.

In short, there may be value dependencies among exchange requests, but nothing in AICE represents these dependencies or acts upon them.

3.1.5 Multiple Destinations

As mentioned above, a single exchange request r_k can include multiple destinations, expressed as a set $\mathbf{d}_k$. A request with multiple destinations arises when IDM or some entity external to AICE wants the same information to be delivered to those destinations.[4] What importance should be ascribed to such a request? How does it related to the importance of delivering information to each of the destinations individually?

In general the importance should be set so that AIC recognizes the value of these deliveries during resource allocation. However, there are two problems in determining this value:

- There is no way to determine from the request whether the value of delivering the information to one of the destinations is independent of the value of delivering it to any of the others. (See section 3.1.4.)
- Even assuming that the values of delivering to the different destinations are independent, the form of the optimization criterion affects the selection of the correct value. For instance, if the optimization criterion in equation (1) is used, the importance value for the request should be the sum of the importance values for delivering the information to each of the destinations. If the optimization criterion in equation (2) is used, *no single scalar importance value conveys the necessary information*; AIC would need to know the importance values for the individual destinations.

Our solution, which is not completely satisfying, is to set the importance of a multi-destination request to be the *maximum* of the importance values generated by considering the destinations individually. In other words,

$$p(w_k, s_k, \mathbf{d}_k, c_k) = \max_{d \in \mathbf{d}_k} p(w_k, s_k, \{d\}, c_k)$$

---

[4] The Information Dissemination Manager (IDM) is an AICE component that attempts to reduce overall network load by (1) recognizing similar information requests from multiple applications and (2) merging them into a single multicast channel request that carries sufficient information to meet all the applications' needs.

The rationale is:

- We assume that the multiple destinations have roughly the same importance, else IDM would not have combined the destinations into a single request.
- Under this assumption, AIC can compute information delivery value according to either (1) or (2), since it knows both the importance value for each destination (they are roughly the same) and the number of destinations.

These two problems suggest that the interface between IPM and AIC should be redesigned in future years to solve these problems. One way would be for IPM to "hold" the complete optimization criterion (e.g. (1) or (2)) and for AIC to be a more generic optimization engine that attempts to maximize that criterion.

3.2 Effects of Policy (Evaluation)

The central question to be answered in an evaluation of the effects of policy is almost certainly: does it make a difference? That is, if we are able to capture the intent of a commander and supply it to some decision-making software, will the results be better than what would have happened absent that intent specification?

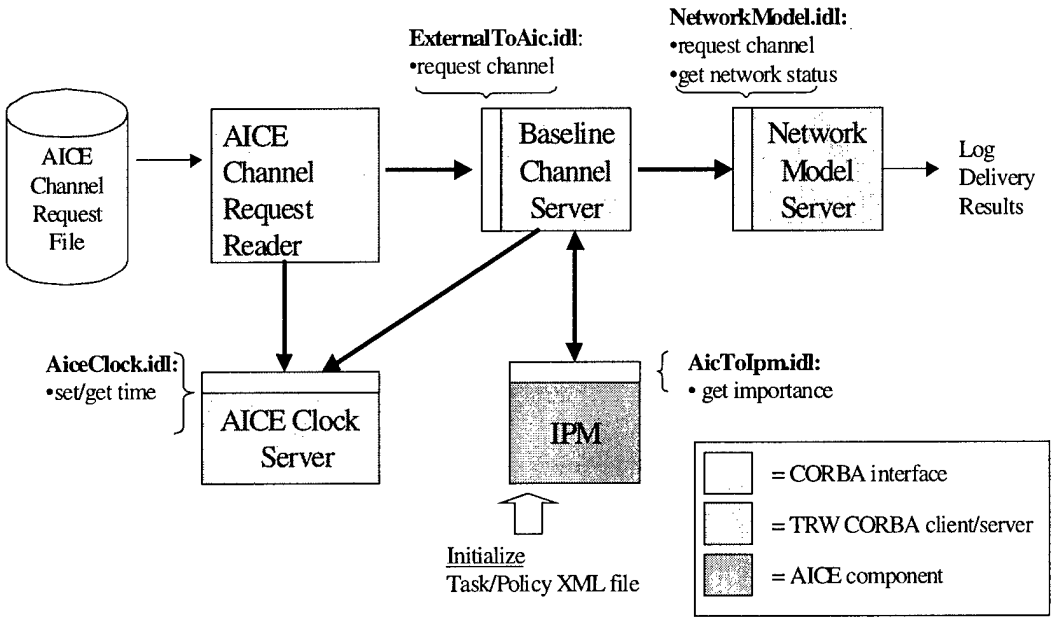

Figure 5. Experimental Configuration

In an effort to answer that question, TRW Reston was asked to perform an integration and evaluation role[5]. Each of the Agile Information Control Environment (AICE) contractors supplied TRW with software to perform their function. TRW constructed a baseline scenario consisting of simple priority assignment, allocation, and network function simulations. For the Information Policy Management (IPM) layer, the IPSO-FACTO software replaced the simple

[5] Report available separately.

priority assignment module to assess the contribution of this layer to performance improvements provided by AICE. The IPSO-FACTO interface to the remainder of the simulation environment was by means of a getImportance call, which would return a scalar in the range [0 ..1] given a channel request specification as described above.

This evaluation environment was subjected to multiple runs with varying forms of policy specification to discern the factors in policy specification that would have the largest effect on performance. Thirteen different policies were constructed, varying across two dimensions:

- The level of detailed information available in the policy, and
- The resolution approach—how different policy statements were combined. (See Section 4.1.2.)

Table 1. Policy designations for 13 tested approaches

Detail: / *Resolution:*	**Only Mission importance**	**Mission importance + task importance**	**Mission, task, and content importance**
Max (subordinate cannot exceed stated value)	L1 (resolution irrelevant)	L2 Max	FullMax
Min (subordinate must meet or exceed)	L1	L2 Min	FullMin
Use subordinate's importance	L1	L2 _0	Full0
Ignore subordinate's importance	L1	L2_1	Full1
Combine with fixed 50% ratio	L1	L2_0.5	Full0.5
Combine with variable ratio	L1	L2_Mix	FullMix

3.2.1 Evaluation Hypotheses

A number of hypotheses were of interest to both us and DARPA. TRW performed an impressive range of simulation runs, each with extensive logging, to provide data for testing these hypotheses.

Hypothesis: Allocation guided by policy will outperform allocation without such guidance.

Discussion: This hypothesis seems almost a given, though upon further inspection, there are a number of interesting questions still to be answered.

First and foremost, what does "outperform" mean? TRW established an independent measure of worth for satisfying a particular channel request for the abstract baseline scenario. Their worth metric took into account the mission that was supported by the channel request, the particular task supported, and reflected the notion that certain types of information typically have more value than others (e.g., targeting information is usually more valuable than logistics). These factors were weighted (mission supported having 10x the effect of the other two characteristics), then summed to form the worth of an individual request's satisfaction. In this evaluation,

"satisfaction" was taken to mean the allocation of a channel for sufficient time and in sufficient bandwidth to complete the transmission. No partial credit for incomplete transmissions was given. This might accurately model, for example, FTP transmissions that provide no value when interrupted.

Absent a realistic scenario, the validity (accuracy with respect to ground truth) of any worth metric is a matter of opinion. TRW also constructed a militarily realistic simulation case derived from the Lantica scenario generated at Ft. Leavenworth. This simulation case gave subject matter experts enough information that independent assessments of information worth were possible. Unfortunately, the complexity of this case also meant that the number of runs performed was substantially reduced, and the data analysis performed similarly reduced. The results presented here, then, will reflect the abstract "baseline" scenario only.

Second, what should be used as the "without guidance" case? A baseline case used a simple first-come, first-served allocation scheme. When resources had been consumed such that a request could not be granted, the request was rejected. When ongoing channels had been completed, resources were freed, and new incoming requests could be granted. Clearly, the more requests that are rejected, the more room for improvement there is in satisfying the high-value requests (assuming a random value presentation sequence). Figure 6 shows two conditions for the simulation, one in which the network is continually overloaded, the other in which the network is continually underloaded. The traces show the cumulative value of:

- Requests that were presented to the allocator,
- Requests that were allocated network resources
- Requests that were rejected outright
- Requests that were pre-empted in favor of a higher value request,
- Requests that were completed.

The underloaded case, even with an unsophisticated allocation strategy, performs perfectly – there is no room for improvement.

Figure 7 shows the completed request worth for all 15 runs for one network configuration (CRW1) baseline. Note that the performance for all cases except the overload is within about 25% of the best possible score. As a result, we will concentrate mainly on the overload cases in our analysis.

Finally, what does "guided by policy" mean? The simplest meaning is that the allocation module attends in some way to the scalar importance value returned by IPM. To the degree the semantics of the importance value for AIC match the semantics for IPM, and that they both match the worth metric used by TRW, the allocation scheme should benefit from the presence of such information.

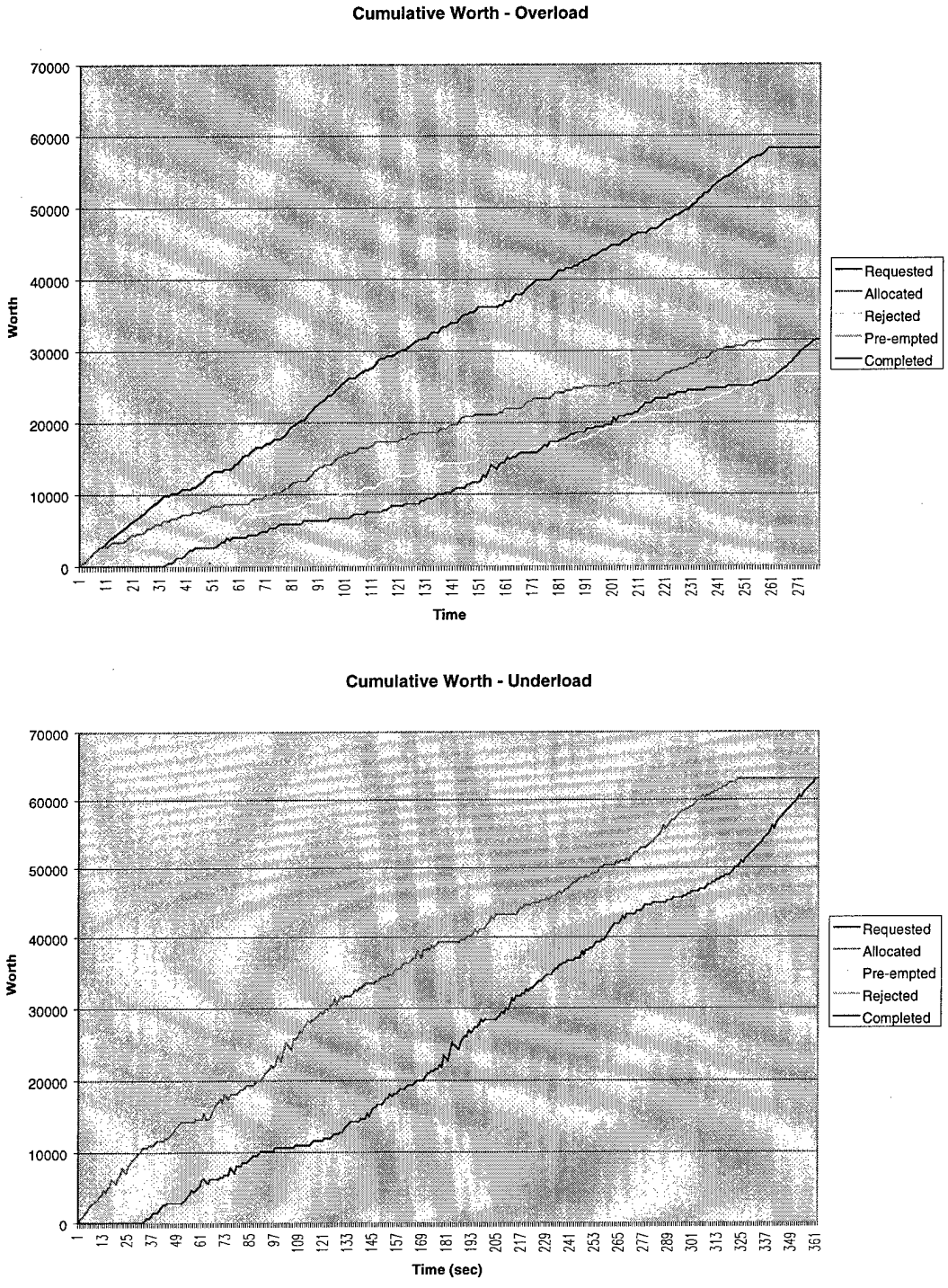

Figure 6. Baseline Allocation Strategy, Overload and Underload Conditions

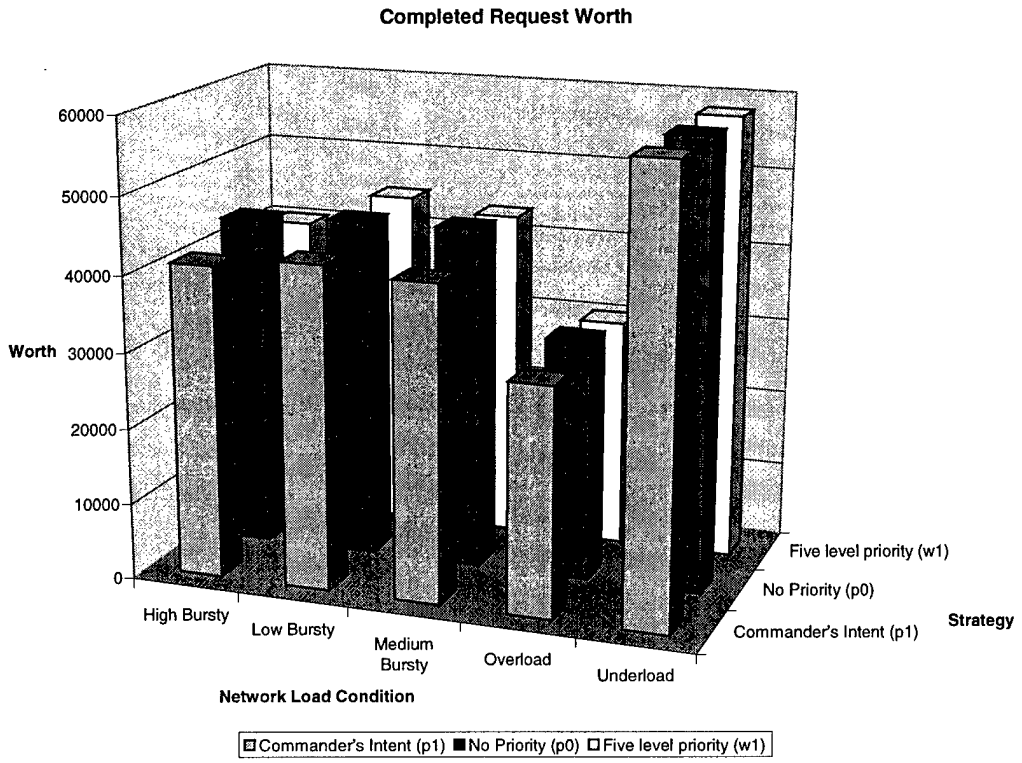

Figure 7. No Policy Results for One Network Configuration

Figure 8 shows that for a majority of cases, this supposition appears to hold. For the case where not having a policy outperforms the with-policy case, the results are nearly identical. Even so, the results are not statistically significant, even at the $P < 0.05$ level. A one-tailed paired two sample t-Test for these six run pairs gives $P(T \leq t) < 0.06$.

Interestingly, for the policy that we expected to perform the best, the difference between its performance and the no-policy case is even smaller. A one-tailed paired two sample t-Test for these six run pairs gives $P(T \leq t) < 0.12$.

One would also expect that a poorly constructed policy would regularly underperform the allocation results uninformed by policy. While we had no cases in which the policy was deliberately contrary to intent, Figure 9 compares the no policy case to a policy which has substantial detail, but only local knowledge of importance. This policy underperforms having no policy at all in a majority of cases. Again, this result is not statistically significant.

Hypothesis: Policies that provide guidance based on mission, task and content information will outperform those that use only mission, or only mission and task information.

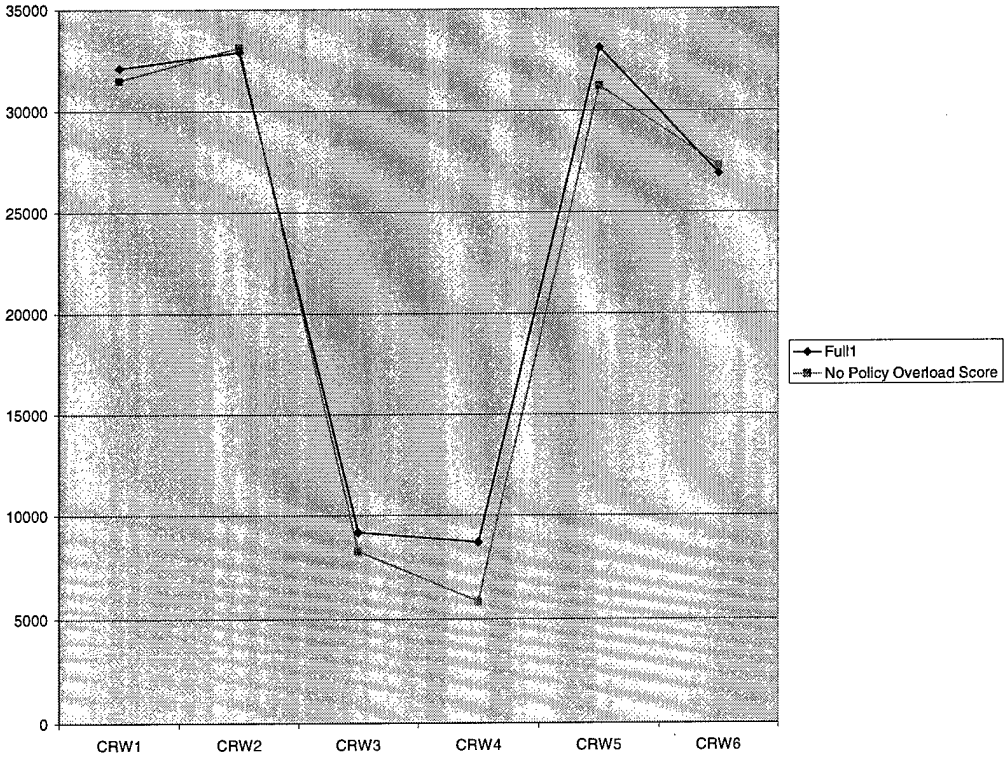

Figure 8. Complete Policy versus No Policy

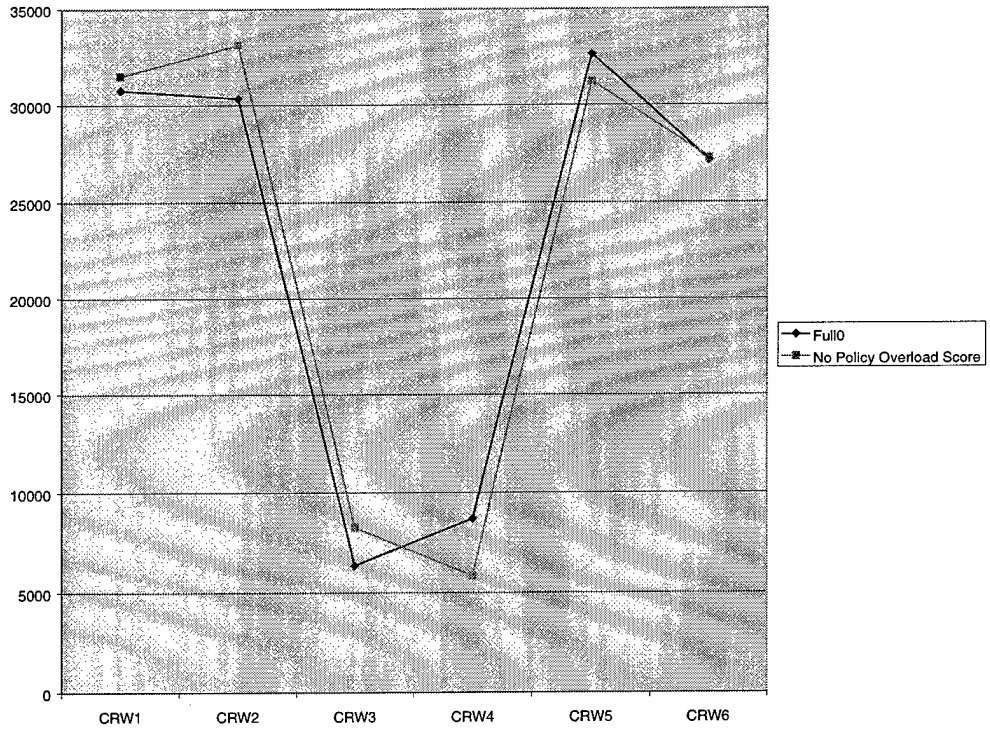

Figure 9. Poor Policy versus No Policy

Discussion: Again, this hypothesis was not supported by the data. In fact, for this hypothesis, not only is the result not statistically significant, the data actually are contrary to the hypothesis. Figure 10 shows that a coarse grained policy (Level1) outperforms a policy with more information (Level2_Mix), which in turn outperforms the full detail policy.

We considered this sufficiently remarkable that we began a more comprehensive data analysis.

Upon closer inspection, it was apparent that the issue was in the semantics of the policy statements: what did they mean to IPM, to AIC, and to the experimental value metric. In fact, the meaning of policy was not consistent across the three entities. In particular, the interpretation used by AIC in resource allocation was not the same as the valuation function used by the experimental team to assess outcomes. At least some of the AIC implementations would pre-empt a lower priority transmission in favor of a higher priority transmission, believing that they could gain additional "points" by having the higher priority transmission active. In fact, using the experimental metric, it can be seen that interrupting a transmission in progress in favor of a (somewhat) higher importance transmission is generally a bad idea—the resources allocated to the incomplete transmission have been squandered.

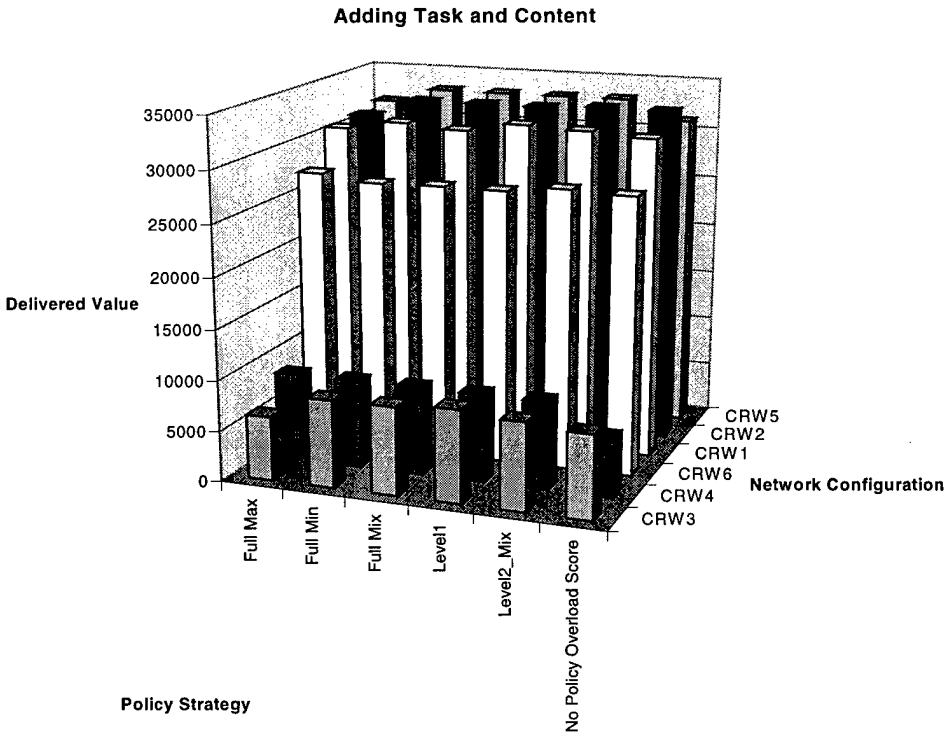

Figure 10. Added information does not improve performance.

Therefore, we have begun analysis of the experimental runs using a proportional partial credit assignment for incomplete transmissions[6]. This might match a periodic report, or a "resumable" FTP transmission. As seen in Figure 11, assigning partial credit results in the detailed, flexible

[6] This analysis was not possible for the Lantica scenario; not all the relevant data was not collected.

policy consistently outperforming the policy-free solution. ($P(T \leq t) < 0.0002$, using a one-tailed t-test for paired samples.)

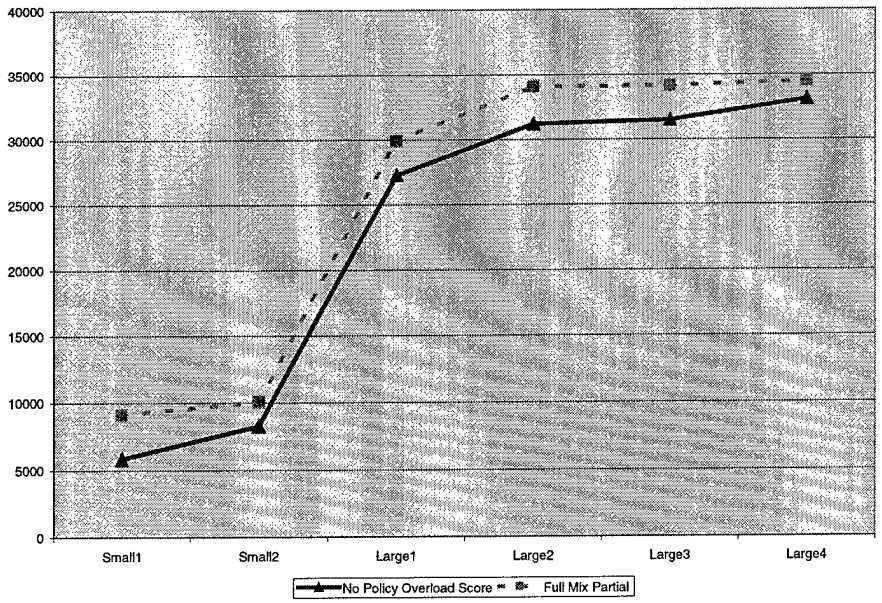

Figure 11. Delivered Value, Complete Policy vs. None with Partial Credit

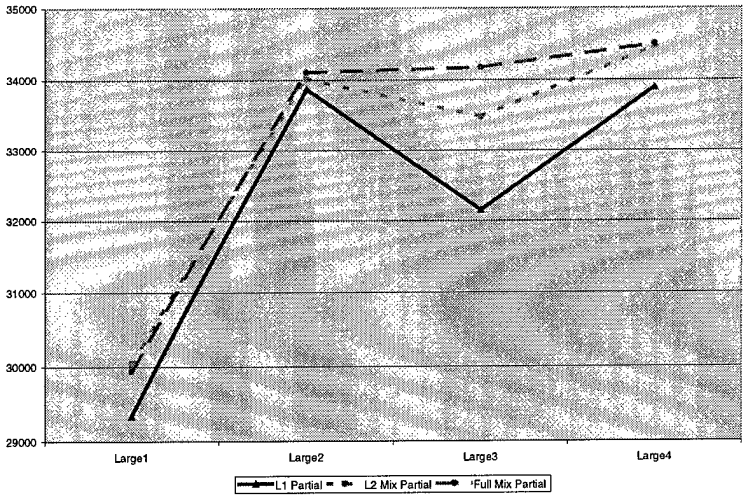

Figure 12. Added information provides value (partial credit).

Given this interpretation of value, the other hypothesis is supported as well. Figure 12 shows that having more detail (Full > L1, $P(T \leq t) < 0.04$) available to the allocation process indeed improves performance.

Conclusion: With a consistent interpretation of "value" across the three entities, the expected results were obtained: policy adds value, and detailed policy outperforms coarse policy.

Section 4
Cross-Echelon Approach to Policy Definition

The syntax and semantics of our formulation of policy, presented in Section 3, are clear and unambiguous. Our evaluation has also shown that the formulation is quite expressive: the formulation can capture a variety of informal statements of commander's intent, resulting in improved value of delivered information. Unfortunately, in the real world, no one individual has enough information to devise an optimal policy. Commanders at high levels have global general knowledge, but insufficient detail to make fine-grained assertions about policy. At lower levels, commanders have immediate and detailed knowledge of their local purview (scope), but have limited knowledge of how that scope interacts with their peers and superiors. It is the combined view of the local commander, his/her peers and his/her superiors that comes the closest to expressing the globally optimized policy statement.

This section presents our approach to cross-echelon policy definition: how the *resolved view* of policy, described in Section 3, is generated by combining policy statements from multiple commanders. Following a brief discussion of human factors considerations in cross-echelon policy definition, we present results from a performance evaluation of this approach for simulated military applications.

4.1 Mathematical Formulation of Cross-Echelon Policy

We describe here how a resolved policy function is derived from a collection of commanders' policies. As explained in Section 3, a resolved policy is a function $p : W \times A \times \mathbf{P}(A) \times C \rightarrow I$ that defines the importance $i = p(w, s, \mathbf{d}, c)$ of a request $r = (w, s, \mathbf{d}, c, u)$, where:

- w is the *owner*—the creator of the exchange request. The set of all possible owners is W.
- s is the *source*—an application entity that participates in an exchange. In a one-way exchange, the information comes from the source. The set of all possible sources is the set of applications A.
- $\mathbf{d}$ is the *set of destinations*—a set of application entities that participate in an exchange. In a one-way exchange, the information goes to all destinations. The set of all possible destinations is also the set of applications A.
- c is the *exchange characterization*—a description of the information content of the exchange. The set of all possible exchange characterizations is C.
- u is the *flow characterization*—a function that defines the owner's quality of service requirements for the exchange. The set of all possible flow characterizations is U.

For simplicity of exposition, we assume here that a request has a single destination d, so that requests are of the form $r = (w, s, d, c, u)$ and a resolved policy is a function of the form $p : W \times A \times A \times C \rightarrow I$ that computes the resolved importance $i = p(w, s, d, c)$. If a request has multiple destinations, the importance of the request is computed by taking the maximum of the

resolved importance values generated by considering the destinations one at a time. The rationale for this treatment of multiple destinations was presented in Section 3.

There are three principal elements in the derivation of resolved policy:

- The *governing tree*, the structure to which commanders' policies are attached.
- *Commanders' policies and importance resolution*—the basic form of a commander's policies and how the importance values that different commanders' policies assign to the same request are combined into a single resolved importance value.
- *Structure of commander's policy*—the detailed structure of a commander's policy as a sequence of policy elements.

4.1.1 Governing Tree

The governing tree (Figure 13) represents a command hierarchy. The nodes in this tree represent tasks in a hierarchical mission plan; they could also represent organizational units or other objects that represent commanders' responsibilities. For each node, there is a commander—a human being—who defines the policy for that node.

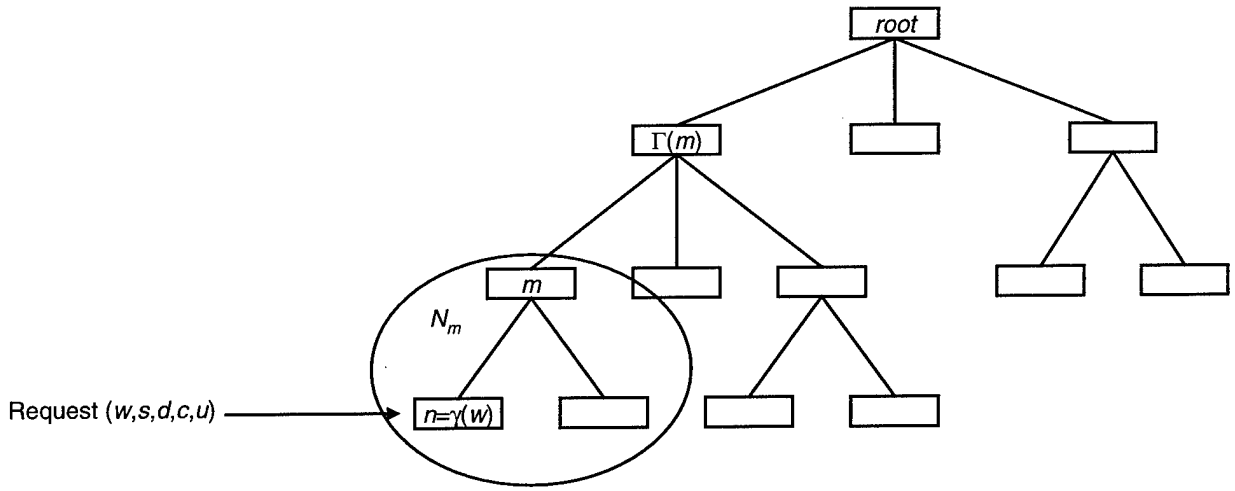

Figure 13. A Governing Tree

The tree is represented mathematically by the set of nodes N and a *governing function* $\Gamma : N \rightarrow N$ that maps each node m in the tree (except a distinguished node *root*) to its parent $\Gamma(m)$. For any node m in the tree, let N_m be the set of nodes in the subtree rooted at m. Then $N_{root} = N$.

As stated earlier, each request r has an owner w, who is the creator of the request. The owner is acting in support of some node (task) n in the governing tree. Node n is called the request's *originating node*. This relationship between an owner and a node is represented by a *supporting function*—a function $\gamma : W \rightarrow N$ that identifies the originating node $n = \gamma(w)$. In a full implementation of these concepts, there needs to be some way of determining the originating node, e.g. by requiring the owner to supply it, by consulting system-maintained contextual information, or by attempting to infer the node from the owner's actions. For now, we make the

simplifying assumptions that request owners *are* tasks and that the owner of a request *is* the task that the request supports: $W = N$ and $n = \gamma(w) = w$.

4.1.2 Commanders' Policies and Importance Resolution

Each node m in the governing tree has a commander's policy that *asserts* the importance of requests originating in the subtree rooted at m. More precisely, a commander's policy for node m is a function of the form

(3) $$p_m : N_m \times A \times A \times C \rightarrow I$$

This function defines the asserted importance i_m^a of a request $r = (w, s, d, c, u)$:

$$i_m^a = p_m(n, s, d, c) \text{ where } n = \gamma(w).$$

If a request originates at node n, each node m at or above n in the governing tree asserts an importance i_m^a for the request. Let these asserted importance values be $i_n^a, i_{\Gamma(n)}^a, \ldots, i_{root}^a$. These asserted importance values may be different, and must be *resolved* (combined) to generate a single resolved importance value that AIC uses for resource allocation.

Resolution is performed from the bottom up, from the originating node n to *root*.[7] Resolved importance values $i_n, i_{\Gamma(n)}, \ldots, i_{root}$ are computed for each of these nodes as follows:

(4) $$i_n = i_n^a$$
$$i_{\Gamma(m)} = \rho_{\Gamma(m)}(i_m, i_{\Gamma(m)}^a)$$

Here, $\rho_{\Gamma(m)}$ is an *importance resolution function* that is supplied by node $\Gamma(m)$ for this request. An importance resolution function is a function $\rho : I \times I \rightarrow I$ that is used to merge importance values as shown above. Examples of resolution functions include:

"MAX"	$i_{\Gamma(m)} = \max(i_m, i_{\Gamma(m)}^a)$	Superior node's asserted importance is lower bound on resolved importance.
"MIN"	$i_{\Gamma(m)} = \min(i_m, i_{\Gamma(m)}^a)$	Superior node's asserted importance is upper bound on resolved importance.

[7] This is a change from the AICE functional architecture as documented in [REF _Ref455295138 \r \h], which describes top-down resolution. While it is reasonable and necessary for the commander to know how his policy is affected by his superiors, we expect the commander to provide us importance values that are accurate when considering his command, not for the entire battlespace. We believe the commander will be able to provide meaningful estimates for this local scope. As a result, the commander must know the effective importance assigned to contemplated information transmission acts at his level, which is the resolved effect of all policy statements subordinate to him, plus his own.

"COMBINE(a)"	$i_{\Gamma(m)} = (1-a)\cdot i_m + a\cdot i^a_{\Gamma(m)}$	Resolved importance is linear combination of superior's asserted importance and inferior's resolved importance, with *combining factor* $0 \le a \le 1$. If $a = 0$, inferior's resolved importance is used; if $a = 1$, superior's asserted importance is used.

The importance resolution function ρ_m supplied by node m can depend on the attributes of the request. In other words, *importance resolution functions are part of a commander's policy.* A commander's policy returns not only an asserted importance value for a request (Equation (3)) but also resolution function:

(5) $\quad p_m : (N_m \times A \times A \times C) \rightarrow (I \times (I \times I \rightarrow I))$

The resolution function supplied by p_m is only used for requests whose originating nodes are strictly below m in the governing tree, as shown in Equation (4).

Computing resolved importance must be fast, because AIC will frequently ask IPM to determine a request's importance. Policy changes are expected to be infrequent compared to importance computations. It is tempting to consider *compiling* the governing hierarchy and associated policies into a representation in which the results of importance resolution (Equation (4)) have been precomputed. It may be possible to improve the time-efficiency of importance computation by using such a representation. The speed-up, if any, will depend on the specific representation chosen for commanders' policies (see below), and our initial investigations suggest that a compiled representation may have significantly higher space requirements that outweigh any speedup. However, this is an area for further investigation.

4.1.3 Structure of Commander's Policy

This formalization of commanders' policies allows a very broad class of policies, including some quite expressive policies, e.g. "give high importance to destinations that are under fire", and some that are clearly ridiculous, e.g., "the importance of a request is inversely proportional to the number of characters in the source name". For our first implementation of these concepts, we have chosen to provide a limited class of policy functions having a specific structure. The class has been defined with the two factors in mind: the ability for human beings (the commanders) to understand and specify the policies, and implementation feasibility.

IPM represents each asserted policy p_m as a sequence of *policy elements* $\langle p_{m1}, \ldots, p_{ml(m)} \rangle$, where $l(m)$ is the length of the sequence. Each policy element defines, for a particular class of requests, the asserted importance of that class and the importance resolution function to be used with that class. A class is specified by a set of originating nodes, a set of sources, a set of destinations, and a set of exchange characterizations. More formally, a policy element p_{mj} is a tuple $(pn_{mj}, ps_{mj}, pd_{mj}, pc_{mj}, i_{mj}, \rho_{mj})$ where pn_{mj}, ps_{mj}, pd_{mj}, and pc_{mj} are *predicates* over N_m,

A, A, and C, respectively, $i_{mj} \in I$, and $\rho_{mj} : I \times I \rightarrow I$. Taken in isolation, a policy element means:

$$\forall n,s,d,c \; pn_{mj}(n) \wedge ps_{mj}(s) \wedge pd_{mj}(d) \wedge pc_{mj}(c) \Rightarrow p_m(n,s,d,c) = (i_{mj}, \rho_{mj})$$

To compute the asserted importance i_m^a of a request $r = (w,s,d,c,u)$, the originating node $n = \gamma(w)$ is determined and the policy elements are scanned in sequence until one is found that matches on n, s, d, and c. The importance value and the importance resolution function in that policy element are then used as the value of $p_m(n,s,d,c)$. More formally, the sequence of policy elements $\langle p_{m1}, \ldots, p_{ml(m)} \rangle$ represents the policy p_m as follows:

$$p_m(n,s,d,c) = (i_{mj}, \rho_{mj})$$
$$\text{where } k = \min\{ j \mid 1 \le j \le l(m) \wedge pn_{mj}(n) \wedge ps_{mj}(s) \wedge pd_{mj}(d) \wedge pc_{mj}(c) \}$$

If there is no such k (i.e. if there is no match for s, d, and c among the policy elements) then by default $p_m(n,s,d,c) = (i_0, COMBINE(0))$, meaning that the commander assigns a *default importance* i_0, and if the request originated at a subordinate node, the resolved importance of the immediately subordinate node will be used. The default importance i_0 is a global parameter whose value is chosen based on experience with actual request streams.

We have defined a particular class of predicates pn_{mj}, ps_{mj}, pd_{mj}, and pc_{mj} for use in our initial implementation. The predicate pn_{mj} is over the set of originating nodes N_m that are at or below m in the governing tree. In other words, pn_{mj} selects some subset of these nodes. In our initial implementation, we specify this subset by specifying a single node m' at or below m in the governing tree. The nodes selected by pn_{mj} is $N_{m'}$, the set of nodes at or below m' in the governing tree. In other words, for any originating node n, $pn_{mj}(n) \Leftrightarrow n \in N_{m'}$.

The predicates ps_{mj} and pd_{mj} are over the set of applications A. Ideally, these predicates could qualify applications according to a number of attributes, including owner organization, geographic location, security level, etc. This kind of capability is provided by the X.500 Directory Service, which has been adopted by Defense Messaging Service [1,2] and by the Lightweight Directory Access Protocol (LDAP) [3,4]. We have decided not to incorporate X.500 and LDAP in our initial implementation due to the engineering effort required. Instead, we are using a "poor man's X.500 directory" consisting of a tree structure that represents a command organizational hierarchy and applications "owned" by organizational units in that hierarchy. See Figure 14. Organizational units can be interior or leaf nodes. Applications must be leaf nodes. Predicates ps_{mj} and pd_{mj} are specified in the same way as pn_{mj}—by selecting a specific node in the command organizational hierarchy. If an organizational unit node is selected, the interpretation is "all applications belonging to this organizational unit or its subunits". If an application node is selected, only that specific application matches the predicate.

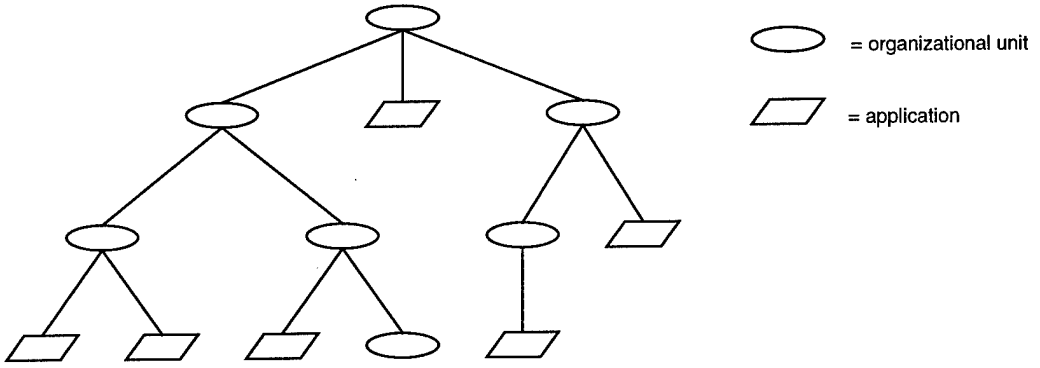

Figure 14. Example Command Organizational Hierarchy

The predicate pc_{mj} is over exchange characterizations. The conceptual approach for specifying exchange characterizations, developed by MITRE [5], is a simple content-type descriptor label, taken from a mission-oriented classification hierarchy such as the following:

```
Information
   C2
      Logistics
      Tasking Orders
   Weather
   Intelligence
      HUMINT
      IMINT
         EO
         IR
         SAR
      MASINT
      SIGINT
         ELINT
         COMINT
```

Given this hierarchical structure, we have again chosen to specify predicate pc_{mj} by selecting a node within the hierarchy. So, for instance, if pc_{mj}=`C2`, exchange requests having an exchange characterization of `C2`, `Logistics`, or `Tasking Orders` will match.

4.2 Usability Considerations in Cross-Echelon Policy Definition

The meaning of asserted importance—How should commanders assign asserted importance when formulating policy? In particular, should each commander use asserted importance to rank a request relative to:

- Other requests that originate within his subtree of the governing tree, so that importance 1 means "most important in my subtree", or
- All requests on the battlefield, so that importance 1 means "most important overall"?

We recommend the former interpretation, since in general a commander cannot assess the relative value of his activities to those of other commanders, and may not even be aware of them.

Giving commanders insight into policy resolution—A commander, when defining a candidate policy element for his node in the governing hierarchy, would like to understand the impact of policies higher and lower in the hierarchy on the importance that IPM computes for exchange requests that match this policy element. The interactions among policy elements at different levels can be quite complex because each element carries four dimensions of qualification: originating node, source, destination, and exchange characterization. Existing elements can overlap the candidate element partly or completely in each of these dimensions. Two requests that match the candidate element may match completely different elements at higher and lower levels, resulting in different resolved importance values.

Despite this complexity, there are ways of giving the commander insight into how his asserted policy is transformed to resolved policy:

- When a commander is creating a policy element, IPM could identify and display policy elements higher and lower in the governing tree that impinge on the class of requests addressed by this policy element. The commander could select particular elements at higher and lower levels; IPM could then compute the class of requests that match each of these elements, and the resulting importance. This approach might also serve as a critiquing function, reminding a commander of characteristics that another commander considers relevant. The commander might then consider these characteristics for inclusion in his/her policy.
- When IPM, AIC, and MetaNet are operating in integrated fashion, the commander will be able to visualize what exchange requests were actually granted, and with what quality of service. It will be possible to show, for any specific request, its resolved importance and its importance with respect to each commander's asserted policy. This will highlight any significant differences among the commanders' asserted policies.
- The same function can be provided for the evaluation of "what if" scenarios, in which proposed policies are evaluated against simulated workloads. Of particular interest are the "what if" scenarios based on predicted load using branches of the current mission plan at various levels of detail. REF _Ref478199198 \r \h Section 5 discusses this further.

The PAC 99 implementation of IPM does not include these capabilities; they are candidates for implementation in future years.

4.3 Evaluation

Our evaluation of cross-echelon policy formulation focuses on the influence of resolution on delivered value of information. We have not formally evaluated our policy representation with respect to clarity, expressiveness, and fit to the intended users and their responsibilities. Informally, we have observed that generating this representation is very labor-intensive. We believe that the policy formulation process can be greatly simplified using a task-based approach as described in Section 5 and Section 6.

As discussed in Section 3, TRW constructed a baseline scenario consisting of simple priority assignment, allocation, and network function simulations to host the experimental analysis. For the Information Policy Management (IPM) layer, the IPSO-FACTO software replaced the simple priority assignment module to assess the contribution of this layer to performance improvements provided by AICE. The IPSO-FACTO interface to the remainder of the simulation environment was by means of a getImportance call, which would return a scalar in the range [0..1] given a channel request specification as described above.

This evaluation environment was subjected to multiple runs with varying forms of policy specification to discern the factors in policy specification that would have the largest effect on performance. Thirteen different policies were constructed, varying across two dimensions:

- The level of detailed information available in the policy, and
- The resolution approach—how matching policy assertions from different governing nodes (commanders) were combined.

Table 2. Policy designations for 13 tested approaches

Detail: / *Resolution:*	**Only Mission importance**	**Mission importance + task importance**	**Mission, task, and content importance**
Max (subordinate cannot exceed stated value)	L1 (resolution irrelevant)	L2 Max	FullMax
Min (subordinate must meet or exceed)	L1	L2 Min	FullMin
Use subordinate	L1	L2_0	Full0
Ignore subordinate	L1	L2_1	Full1
Combine with fixed 50% ratio	L1	L2_0.5	Full0.5
Combine with variable ratio	L1	L2_Mix	FullMix

In each of these cases, the policy information available at any node in the governing tree was an accurate reflection of at least some portion of commander's intent. That is, this set of experimental trials did not deal with the issues of sensitivity to differences in competence, or susceptibility to sabotage.

There was another important difference that this set of experiments did not address in depth, namely marked differences between commanders at different positions in the governing tree. The commander's intent statements that were used in this set of experiments were nearly homogenous - that is, all commanders agreed on ground truth, and had complete visibility into the needs and intent of other commanders. Said another way, there was essentially only *one* commander's intent, which was applied globally.

In an effort to create heterogeneity that could reveal differing performance across the 6 resolution strategies, we artificially introduced a local scope into each commander's statement of policy. At the top level of the governing tree (Level 1), policy included statements regarding the importance

of requests in terms of the mission importance. That is, a request was assigned importance proportional to the importance of the mission it supported. At Level 2, the policy statements dealt with the importance of tasks within a given mission. At Level 3 and below, policy statements dealt with the content (exchange characterization) of the request.

The valuation function used by TRW to determine the value of delivered information (satisfied requests) weighted these factors 10:1:1 (mission importance contribution is ten times task importance contribution, which is the same as content importance contribution). As a result, over 80% (5/6) of the available information to guide allocation was present using only Level 1 policy.

4.3.1 Expected Results

With this structure in mind, we will examine the expected behavior of the various resolution strategies. The optimal strategy (by design) should be a mixed resolution approach, wherein Level 1 commanders suggest that their ratings be given substantial weight (5/6) when combined with their subordinates. Level 2 commanders would assert that their importance ratings be combined equally with their subordinates, and Level 3 and below commanders would say to use the rating of my subordinate, if one exists, else use my ratings. For leaf nodes in the governing tree, of course, the resolution strategy is immaterial—there is nothing with which the ratings need to be resolved.

Short of this optimal strategy, (effective) policies formed by resolving (asserted) policies would be correct to the extent that they emulated this combination. Consider policy as resulting from the addition of several policy "vectors". To the extent that the policy vectors used point in the right direction and have the right magnitude, the effective policy is correct. The other components of the effective policy are unknown, and assuming independence can be treated as random with respect to the correct policy vector.

Thus, a Level 1 policy (one that uses only the information available at the top level of the governing tree) would still perform quite well—it has 5/6 as much correct guidance as a complete policy. Any policy which uses a Combine(1) resolution approach results in an effective Level1 policy—the subordinates ratings are ignored.

A policy which mixed ratings equally (Combine(0.5)) would use all the available policy information (mission, task and content), but weight them incorrectly (2:1:1). This incorrect weighting would result in a policy that had 66% correct information, with the correctness of the remainder unknown.

Of the remaining policy resolution approaches, Combine(0) has the effect of weighting the importance value at the supported node (for that request) in the governing tree at 100%. Note that if all requests were in support of Level 1 tasks, this would reduce to an effective Level1 policy as well. For this experiment, however, most requests supported tasks at Level3 or below. Thus, the Combine(0) policy used content (exchange characterization) information alone to determine importance.

Max and Min, when used globally, serve to create a less discriminating policy, since the effective range of importance values that can be asserted is reduced.

We anticipate that in practice, Max, Min, and Combine(0) will be exception cases, asserted by any given commander with respect to very narrow regions in the request space.

4.3.2 Actual Results

Figure 15 shows the relative performance of the various approaches described above for a policy with importance assertions at all levels of the tree. Again, the results using a valuation metric in which no partial credit is given are not compelling. In fact, in many cases, having no policy at all provides better performance than a policy that is (at least partially) correct. As previously discussed, this is because pre-empting existing channels due to small differences in importance is counter-productive—the pre-empted channel's resources have been wasted.

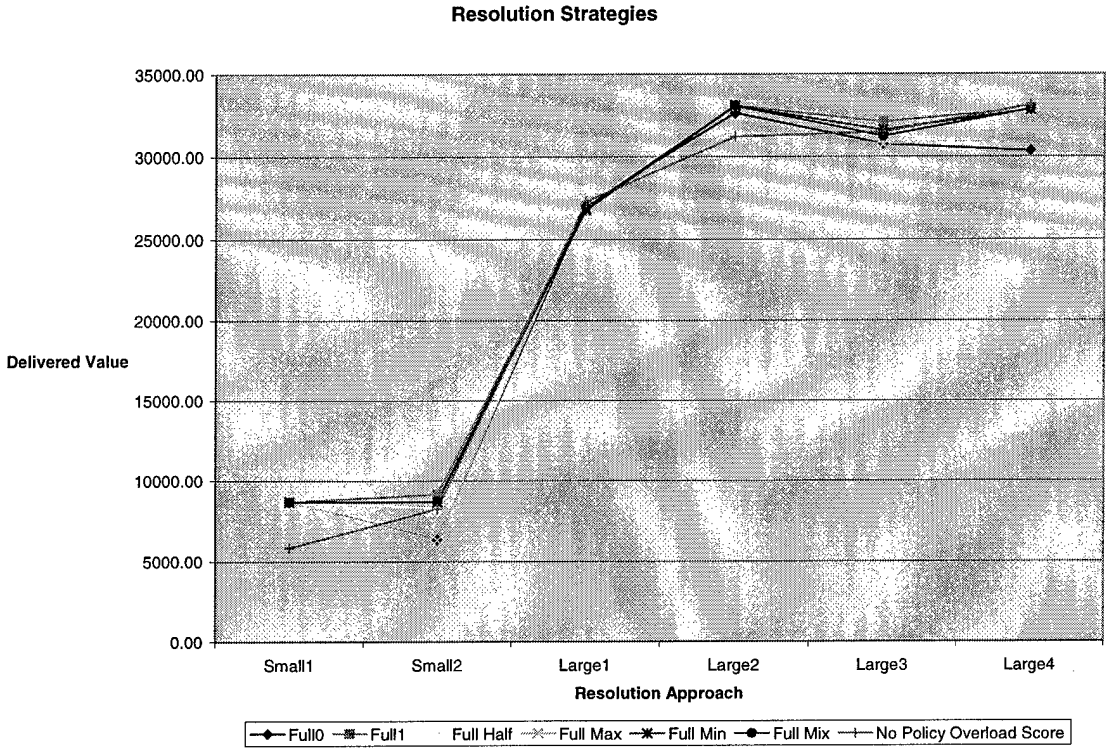

Figure 15. Performance of Resolution Strategies on Full Information

We have computed delivered value scores using a proportional linear credit assignment for approximately 30 of the simulation runs. These data are in accord with our expectations, again emphasizing the necessity for consistency in the interpretation of policy across IPM, AIC and the experimental metrics.

The proportional linear credit assignment assumed that each transmission would take 120 time units (derived from the underload case), and gave credit for preempted channels using

$$\delta = \frac{(\tau_\zeta - \tau_\alpha) \times v_{\max}}{120}$$

where $v_{\max}$ is the value that would have been given had the transmission completed.

Statistical analysis was performed on the total delivered value metric using the partial credit modification, and a one-sided two-sample paired t-Test was applied across the six network configurations for the continuous overload case to test the null hypothesis.[8]

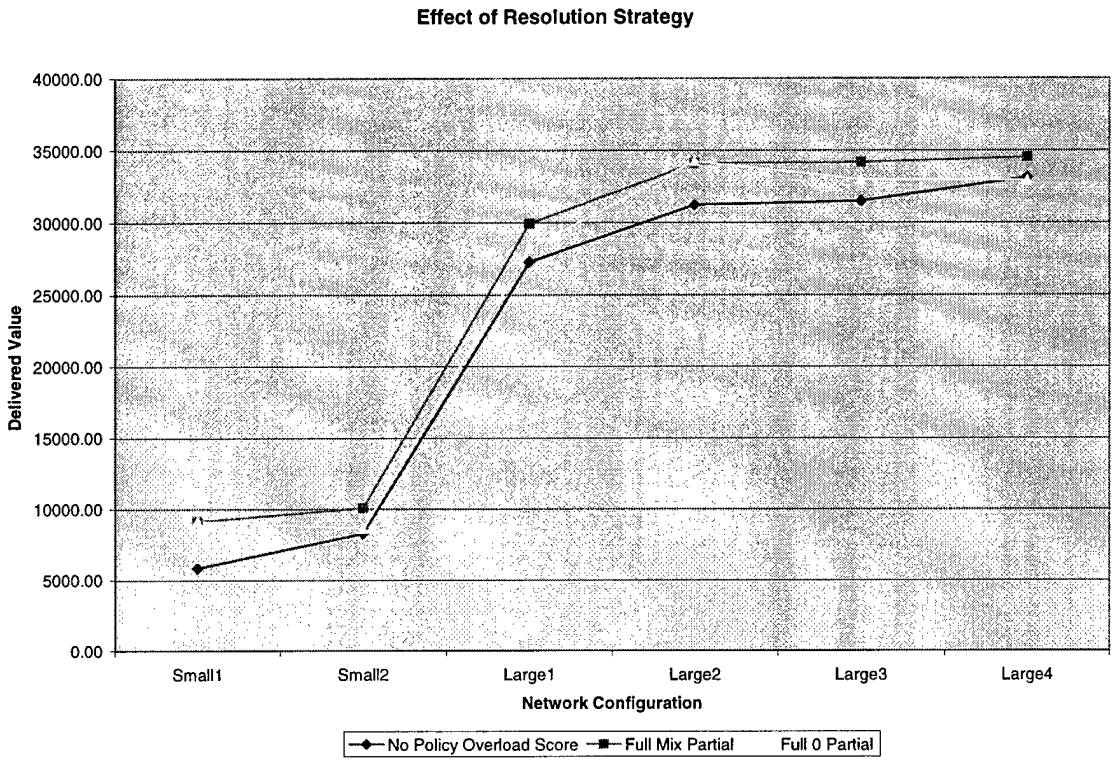

Figure 16. Two Resolution Strategies Compared to No Policy

Figure 16 shows the results for two widely disparate resolution strategies (mixed and "Ignore me, use my subordinate") over the six network configurations for the overloaded case. As expected, the using the full tree of information with a mixed resolution strategy (Full Mix policy approach) outperforms both the no policy case ($P(T \leq t) < 0.0002$) and the full tree of information with a fixed "use my subordinate" (Full Combine(0)) approach ($P(T \leq t) < 0.02$). The Full 0 approach also outperforms the no policy case for a majority of the network configurations ($P(T \leq t) < 0.03$).

4.3.3 Conclusion

With a consistent interpretation of policy across the three entities (IPM, AIC and the experimental metric) it is apparent that the selected resolution strategy has a significant effect, and is in keeping with our predictions.

---

[8] The null hypothesis would have been rejected at the $P < 0.05$ level for the two-sided t-Test as well.

Section 5
Future Directions

5.1 Decision-Aiding for Optimizing Priorities in Complex Domains

IPSO FACTO is only one instance of a class of decision-aiding systems that help systems perform according to human priorities within complex domains. And in its current state, the system is only a first step at that. There are many other domains to which we could apply these techniques, but to make a truly effective system, several research problems would first need to be explored. This section presents a few broad future directions for IPSO FACTO, and the next section discusses in more detail one direction we are especially interested in exploring—the task-based creation of policies.

Other computational optimization systems are designed to be almost completely automated. As discussed earlier, our system is designed to allow a human to be an integral part of the optimization process—a mixed-initiative optimizer if you will. As a result, a user can actively guide the behavior of the system as optimization occurs.

In order for this to work, the user and the system must share a common representation of *context*, one that allows the user to intuitively enter what's 'good', and at the same time, one that the system can use to guide computation. Users must communicate their desires in more detail than the verbal shorthand humans use with each other, but the system can't require detail so complex that the system becomes unusable.

Interesting questions arise, then, for other applications in other domains. What is 'good' in a given domain? Assuming that the domain is halfway complex, 'good' is probably dynamic and context-dependent. So what are the various ways of representing context?

For a large class of domains, we believe that the chief factor in determining what's good is *tasks*—what's good depends on what the users are doing at the moment and what they plan to do in the future, their current and projected activities. What, then, are the kinds of tasks that users commonly perform in this domain? For each task, what kinds of resources are helpful in accomplishing that task? Requests for resources associated with high-priority tasks would be, by association, important to satisfy.

Because there will be multiple tasks active at any given time, with multiple levels of priority, some means must exist to combine multiple statements of goodness into a single, albeit complex, guidance statement. For a large class of applications, we believe that this guidance statement takes the form of an optimization function. What details, or independent variables, should this function take into account? For a given domain, what details are meaningful? How can the system eliminate unnecessary detail? How can the system recognize that the selected detail is not meaningful, or that the eliminated detail is actually necessary?

For example, consider a mixed-initiative optimizer for the domain of target valuation in air campaign planning. A standard abstract task for an air campaign plan is to *Gain Air Superiority*. Factors in determining the successful performance of this task include number of operable

airstrips, fighter aircraft, SAM sites, radar sites, etc. Knowing that the task of *gain air superiority* is active provides a context for determining what details are meaningful and which resources are necessary. Namely, until this task is achieved, destroying SAM sites and reducing the number of operable airstrips (themselves subtasks with their own context) are a few of the actions that will have higher priority. But these contexts may yet combine with other details that increase or decrease these base priorities. For instance, perhaps the opposing force is composed of mainly V/STOL aircraft, so that reducing operable airstrips would be ineffective, and allocating resources to this task would thus have low priority. Or perhaps only local air superiority is required for a short period of time, a storm front is approaching, and the opposing force has no foul weather attack capability. As a result, this task would require few resources.

Interaction between the contexts of multiple tasks and goals add even more complexity. For example, the opposing force may have interspersed SAM sites among civilian areas, so that collateral damage metrics become an extremely important detail that compete with the goal for air superiority. Expensive precision-guided munitions may be required, or a standard air munitions attack may be completely unfeasible.

These are some of the interesting issues involved in applying the IPSO FACTO approach to other domains. As mentioned in Section 2, there are many straightforward applications just in terms of resource allocation. However, there are many extensions that can be made to the base approach as well.

5.2 Forecasting and Visualization

One extension to the IPSO FACTO approach includes providing better support for forecasting and visualization. Forecasting can be seen as the computation of the predicted results of enacting certain policies or performing certain tasks. Visualization can be seen as the methods the system uses to meaningfully convey those predicted results back to the user.

In the current version of IPSO FACTO, users can forecast by creating multiple versions of policies and submitting them individually to AIC in an off-line process, and manually comparing the results. This is essentially what the experimental evaluation of IPSO FACTO entailed. A system like IPSO FACTO would be more usable and productive if the user could easily and dynamically adjust portions of the policy or plan and immediately see the results of those changes.

For visualization, the ability to explore different aspects of predicted results would be especially useful, more so than viewing a single global impression about how well a policy works. For instance, consider the target valuation application again. Yes, it would be nice to know in general how well a specified prioritization scheme contributes to achieving air superiority. But it would be even more useful to look at the predicted world state more closely, to take different 'slices', considering factors such as, how would this affect the 3rd Division? What tasks will be supported and which won't? What geography, what personnel will be affected? The system may have generated a highly optimized plan that is unacceptable because the user didn't provide enough constraints.

5.3 Monitoring

There are two types of monitoring we currently envision adding to IPSO FACTO. The first is monitoring the progress of the current situation as compared to the forecasted situation. When the current world state begins to deviate from the predicted state in some substantial way, the user could be notified and adjust the current policies accordingly.

The second type of monitoring is for critiquing the forecasts themselves. As discussed above, a forecast may possess characteristics that make it unacceptable. IPSO FACTO could work to recognize some of the characteristics automatically and bring them to the attention of the user.

5.4 Resource Costs

In its current version, IPSO FACTO's focus is on delivering as much high-value information as possible. In tests, we received optimization points for doing this; our goal was to maximize the number of points we received. However, this view is too narrow. Practically, optimization must consider *cost* as well as benefit. IPSO FACTO currently assumes that resources are free in its domain, but in reality, continually loading the network to maximum capacity may have costs (e.g., risk of detection, risk of information compromise, reduced adaptability). Or the system may be able to expand the network by renting expensive satellite time, thus maximizing the information delivered, but at some point this option isn't worth the expense. Similarly, in the air campaign planning domain, the commander may be able to address all 512 targets if he calls up 4 wings from reserves and brings them overseas, but would it be worth it?

In future versions of IPSO FACTO, then, we need to explore the best ways to develop objective functions that explicitly account for both benefits and costs.

5.5 Adversarial Reasoning

We could also extend IPSO FACTO to assist users in finding possible weaknesses or critical points in their policies. For instance, once the user has created a policy (or mission plan), the system could look for areas caught up in a broad web of dependencies, having a single point of failure, or areas where more resources would be required than would likely be available. One way IPSO FACTO could accomplish this is by analyzing the policy in terms of a valuation space, searching for points where a small change would result in a huge change in outcome. Knowing these critical points would allow users to modify their policy to make it less susceptible to interference, more robust.

5.6 Quality of Service and Information Characterization

In IPSO FACTO even now, a huge gap divides the quality of service and information characterizations we wanted to employ in our policies, and the kinds of guidance that AIC could handle. For instance, we defined the FRIARS quality of service dimensions—freshness, reliability, initiation-time, accuracy, resolution, and scope—but the only dimensions AIC was really concerned with were bandwidth, accuracy, and loss ratio. We would like to discover how implementing our quality of service dimensions would actually affect information allocation, as

well as explore new dimensions, like *privacy*. We would also like to expand our dimensions of information characterization (currently based on owner, source, destination, and content type). What other attributes of information are important to consider when determining priorities?

In addition to expanding characterizations for information allocation, we would also like to explore what similar characterizations can be made in other domains—how does the concept of quality of service and information characterization map to new domains? For example, in the air campaign planning domain, quality of service may map to the concept of measures of merit. What dimensions are important in determining how well an action against a target supports a given task? An air campaign planner might want to compare the tradeoffs between functional vs. physical effects (e.g., power plant destroyed vs. power turned off), duration of effect (power turned off for one week), resource cost (number of munitions spent, amount of fuel burnt, number of personnel lost), and amount of collateral damage, among others. Information characterization may map to target classification. What kind of target is it—SAM site, radar site, divisional headquarters? What aspect of the enemy system does the target affect—leadership, communication, infrastructure? Is the target protected or unprotected? This could determine whether a covered or uncovered strike should be used. Is the target in a civilian area or an isolated area? This could determine whether precision or non-precision weapons should be used. Does the target require any level of covertness? This could determine whether stealth aircraft, or even a covert ground attack should be used.

This section has discussed some interesting possibilities for extending IPSO FACTO. In the next section, however, we discuss in detail an extension we believe to be the most crucial towards ensuring that IPSO FACTO is usable and accepted; the task-based creation of policies.

Section 6
Task Oriented Policy Specification

At the outset of this effort, we made the claim that commanders and their staffs should not specify policy. Rather, they should do what they do best: plan and execute missions—that is, sequences of tasks to achieve some set of goals set by their superiors. The *derivation* of policy (or, as some have termed it, a parallel information plan) from that mission plan can be an automated procedure if the right information is captured prior to and during mission planning. We were concerned that adding policy specification to the commander's already heavy workload would prove untenable. Our experience with manual, direct specification of policy during the first year of the AICE program reinforces this view. In this section, we discuss the future direction we consider most critical to the success of future applications of this technology: task-oriented policy specification.

6.1 Motivation through Evaluation data

We would, of course, like to eventually evaluate IPSO FACTO's policy creation tool with real military commanders in realistic situations. Such evaluation hasn't been our focus in this initial year of research. However, based on our own experience, we think we know what commanders would say about the tool. They wouldn't care much for it.

Why? Even though the resulting policy enables more control over automated resource allocation and improved allocation performance, the policy itself is complex, and thus time-consuming and error-prone to generate. In other words, creating a policy was tedious and frustrating for *us*, and we built the tool. We can only imagine how the typical commander would feel about being asked to create this type of policy in the midst of coordinating a battle. The following sections present a few illustrations of a policy's complexity.

6.1.1 Number of Components

In order for TRW to run its evaluation tests, we put together two major policies corresponding to two different scenarios, an abstract case and a military-specific case (the *Lantica* scenario). One factor to consider when looking at these numbers is that both scenarios were fairly straightforward, hypothetical situations. In an actual planning situation, a policy is likely to grow even larger and more unwieldy.

The final version of the **abstract** scenario contained the following components:

- A hierarchy of 16 tasks were distributed across 3 levels.
- The hierarchy's root task contained 5 policy elements relating to mission-level worth.
- Each second level task contained 2 policy elements relating to task-level worth.
- Each third level task contained 14 policy elements relating to content-level worth.

The abstract case thus contained 183 individual components, with a task to policy element ratio of 16:167, or approximately 1:10.5.

The final version of the **Lantica** scenario contained the following components:

- A hierarchy of 39 tasks were distributed across 4 levels.
- The hierarchy's root task contained 6 policy elements relating to task-level worth.
- Second level tasks contained varying numbers of policy elements relating to task-level worth, ranging from 3 to 13.
- Each third level task contained 12 policy elements relating to content-level worth.

The Lantica case thus contained 243 individual components, with a task to policy element ratio of 13:68, or approximately 1:5.

6.1.2 Time

Because each policy contained so many components, entering a policy into IPSO FACTO required a relatively large amount of time. Even after calculating what the appropriate values, resolution functions, and policy elements should be beforehand, it took us approximately 6 to 8 hours to create the actual policy with IPSO FACTO. Modifying existing policies wasn't much more efficient – we twice made major changes to the abstract scenario, which each took approximately 4 hours.

As an example, consider a commander who changes his or her mind about the importance of an information type in the current policy. Weather data sent to the firing batteries of II Corps has become more important, and the commander wants to increase its priority from 0.53 to 0.72. In order to implement this change, the commander must find each policy element that represents this information type in the task hierarchy and modify its importance. As seen in the previous section, finding specific policy elements in a given policy involves searching through a large number of components, so that even minor changes can begin to require a major amount of time.

Therefore, as policies grow larger and change more quickly, we believe the time needed to create and maintain those policies will grow to the point where, even though the benefits are great, the cost-to-benefit ratio will be too high for direct, manual methods of policy management to remain feasible.

6.2 Approach

A task-based policy creation approach reduces the commander's workload by leveraging work, which he already would do to create a mission plan, to form an associated, consistent information policy specification. Unfortunately, the task of developing such a specification is a difficult one. Today's battle plans are paper documents, while near future applications for battlefield planning are likely to take the form of diagrams and electronic text. Such representations are neither rich nor formal enough to support Information Policy generation. Formal planning representations (e.g., PSL) would be more than adequate for our needs, but such a representation would likely demand detailed plan specification work. Thus, asking a commander to use such a representation might impose even more burden than asking him to create information policies from scratch.

In future work, we will address this problem by creating a "template-based" plan specification system that allows implicit specification of the associated information policy at low cost. The core notion of this approach is the realization that there are regular structures in the tasks and procedures performed in most domains. A 'task template', therefore, represents an encapsulated set of such behaviors—perhaps including alternatives—which is known as a method for accomplishing domain goals. Much of the task-based research in Human Factors concerns itself with tasks at a very primitive level (e.g., 'discriminate two values', or 'activate switch'). In contrast, our task templates are single, familiar tasks at higher levels of aggregation (e.g., 'Attrit Enemy Forces', 'Reconnoiter Area', etc.). These templates are often, in fact, comprised of sub-tasks at lower levels of aggregation.

As Boy points out [6], the naming or labeling of tasks always involves a process of 'rationalization' or abstraction toward a 'template' that leaves the final customization of the behaviors for the actor at runtime. This rationalization is precisely what makes fully automated planning systems difficult to construct, but the overall familiarity of the labeling and organization of "tasks" also makes it a highly natural way to communicate about activities in the domain. A growing realization of this fact is seen in continuing efforts on the Universal Joint Task List (UJTL) [7] and Air Force Task List (AFTL) [8]. As will be described below, we will use a vocabulary of such tasks as the basis for an information policy specification tool. The commander will be able to select task templates familiar to him, instantiate, specialize and prioritize them to the current situation (including assigning them to specific actors), and compose sets of tasks in sequential, parallel, conditional or iterative orders to represent his mission plan. Since each task will have associated with it the information profiles that the actors need to accomplish the task, the resulting 'task model' will convey a significant portion of the commander's information policy through the relative priorities of the tasks themselves. In addition, the task model will contain a representation of the way in which the information is needed to perform this task. Thus, for example, a targeting task may require rapid updates on position, while a force estimation task might be performed with far less frequent updates.

Figure 17 defines our concept of a task template. Task templates may contain a structure of sub-templates, which must be organized via functional decomposition—each task or goal must be decomposable into alternate methods of achieving that goal. Templates will exist in complex relations to other templates, i.e., their "Surroundings". For example, hierarchical decomposition is the super-sub task relationship, used to represent the way a task is broken down into a set of subtasks, and will frequently also represent the delegation of tasks within an organizational hierarchy. By asserting the parent task template, the system automatically knows some information about the child templates that might or must be present whenever the parent is. Flow of execution represents the relationship of a task template to its peers and is useful in managing the sequence of tasks and their associated information needs.

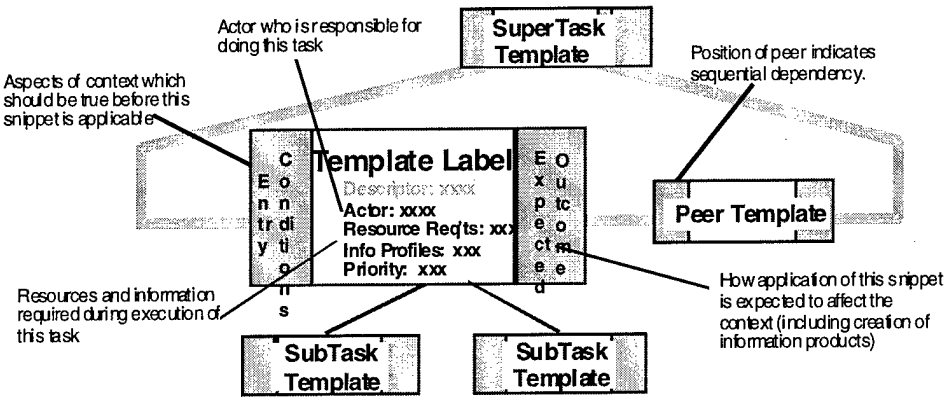

Figure 17. Architecture and Components of Task Templates in IPSO-FACTO

The second requirement of this task model is that the tasks represented must be drawn from the way operators think about their domain. That is, the labeling conventions used by the interface and the system must be the same as those used by the human operators. 'Tasks' may be drawn from training materials, operator interviews, etc., but the resulting model should be intuitive and easily readable by any experienced operator. The work performed to date on the Universal Joint Task List and Air Force Task List will likely provide excellent starting points for such a task model.

Finally, each task template available in the representation must have associated with it a set of information profiles that will be activated whenever that task is activated. This knowledge is the primary source of IPSO-FACTO's ability to automatically extract an information policy via interpretation of a mission plan. The information requirements knowledge will be described in more detail in the next section.

6.2.1.1 Template Capture and Creation

The creation of task templates, and the identification and prioritization of their associated information needs, is critical to the success of the IPSO-FACTO approach. Klein Associates have developed and extensively used Cognitive Task Analysis (CTA) techniques to identify user tasks and information needs in complex, high criticality decision-making domains—including military command and control. Aptima, Inc. has developed techniques for using CTA, called Applied Cognitive Task Analysis (ACTA) and used them in multiple analyses of real-world domains. A typical ACTA approach involves the use of three parallel knowledge acquisition techniques designed and proven to provide converging evidence on the task and knowledge structures inherent in users' thinking about the domain. These are described in Table 3. Klein Associates' technique then involves the use of a technique called the Cognitive Demands Table, whose rows and columns structure has been developed through extensive prior work, to aid system designers in collecting and organizing the wealth of verbal report data collected through prior interviews, and to assist in identifying common themes and priorities across user reports.

Table 3. Convergent interview techniques in the ACTA approach.

Technique Title	Interview Description
Task Diagram	Interview built around decomposing the highest level user tasks progressively into their sub-tasks.
Knowledge Audit	Uses a broad knowledge base from the literature to guide an interview about the important informational cues and strategies used to perform the task.
Simulation Interview	Walks the user through a specific scenario (i.e., 'simulates') to first identify key events and decision points, and then probe them for information cues and decision strategies.

6.2.1.2 Linked Information Requirements Representation

While Klein Associates' approach has been proven a superior method of knowledge acquisition, its results are textual and are therefore not yet useful for the task manipulation methods we wish to employ in IPSO-FACTO. Our team has been working on representing and reasoning about the link between a user's tasks and the information required to perform those tasks for more than ten years. In particular, Aptima, Inc. has extensive previous experience working with Klein Associates and using the results of their ACTA process to develop more formal models of user tasks and information needs. Aptima's approach begins by translating the ACTA Task Diagram and Cognitive Decision Table outputs into Operator Function Models (OFMs). OFMs are hierarchical, containing Nodes at the top and intermediate levels representing major operator functions to achieve overall mission objectives, Tasks at the lowest levels consisting of a series of user actions, and Arcs representing events (both expected, e.g., task completion, and unexpected, e.g., denial of a resource) which can initiate or interrupt tasks and nodes.

Recent work by Aptima [9] has pioneered techniques for combining ACTA and OFM into a shared analytic tool called Cognitive Function Modeling (CFM). This technique incorporates more of the findings from the ACTA analysis into the OFM models to highlight those tasks and functions that will require complex operator decision-making or other cognitive activities. CFM complexity is derived from a combined set of scales for estimating the cognitive demands of different subtasks based on dimensions such as situation awareness, planning/resource allocation, meta-level skills and teamwork factors. Section 6.3 details a particular thrust of this work aimed at using information usage characteristics to aid in the specification of task templates.

In addition, Honeywell has created and implemented a task-to-information association representation that is now flying on the Rotorcraft Pilot's Associate (RPA) program [10, 11, 12]. RPA used a vocabulary of Information Elements (IEs) which were simply names we applied to the different types of information available in the cockpit. During some contexts (specifically,

when some tasks were active) a subset of these IEs could be needed or required, thereby making them Information *Requirements* (IRs) for those tasks. To declare what information was needed, it was sufficient to declare the set of tasks which were currently active, because each task had linked to it a list of parameterizable IRs. IR needs could be satisfied by one or more Presentation Elements (PEs), conveying the IEs that were needed. PEs were organized through rules defining the legal combinations in which PEs could be presented.

To ask for IEs, it was not enough to simply state that a task needed (or that a display provided) "heading" information, however. Instead, we used a descriptive syntax which was akin to QoS metrics. This syntax consisted of statements about the need for particular values of scope, resolution, bandwidth, importance and controllability.

Each parameter had a definition and heuristics (described in [11, 10]) for mapping the range of possible needs and display types into a 10 point, normalized scale for that value. Because the same scales were used for both information needs descriptions and for information conveyance methods, it was trivial to compute the degree of "match" between a need and a conveyance. The match computation could then be weighted by importance or priority values for the information needs within a task and by the relative importance of the task compared to other tasks to provide a metric for human cockpit display management.

Each of these representations has particular strengths that are useful for the AICE domain. We will therefore use a combination of the OFM representation developed by Aptima, and the IE-PE representation developed by Honeywell as the basis of our task templates and task model building syntax.

6.2.1.2.1 Embedded Objective Information

The problem of determining the degree of match between information needed and information provided is the same for AICE as it was for RPA. We believe that these properties hold in other application domains for other resource types as well:

- The resources needed for a given task can be largely specified a priori.
- The ability (or lack thereof) to satisfy a particular resource need will have a variable effect on task outcome, depending on the criticality of the resource need.
- Resource needs can often be satisfied by multiple different resource options.
- The resource option will satisfy the need to varying degrees, according to the "match" of resource and need.

Tasks will have associated with them a list of information needs which should be fulfilled whenever that task is active. While this list will be specialized depending on its location in the mission plan and will be modifiable by the commander (or other user), the goal is the creation of information needs lists which are accurate enough to eliminate, or at least greatly reduce, the need for this sort of direct communication about information needs. Similarly, tasks will have associated priorities when executing; we believe that the task templates can assume reasonable defaults for relative priority, derived in large part from complexity metrics from CFM.

Information profiles within these tasks may have relative priorities as well; these complexity metrics will assist in the prioritization of information profiles within tasks.

While RPA was concerned with communication from a display into the commander's mind, AICE generalizes this concern to communication from an abstract source to receiver. In future work on information transfer, we will define QoS needs and capabilities in keeping with the more generalized problem. We have constructed an initial list of QoS parameters to be encoded with each information needs profile stored with a task template in our representation:

- *Freshness:* How old is the data; how fresh does it need to be?
- *Reliability*: How certain is the data to get through to its receiver complete and intact?
- *Initiation Time:* Can this data be provided (ready to initiate viewing) by a certain time? How critical is it that the information be provided at that time?
- *Accuracy:* How accurate is the data with regards to world truth? How accurate does it need to be? (We note that it will be extremely rare that true accuracy can be known at time of use. Instead, we expect to operationalize this parameter with regards to intel quality ratings or similar quality ranking information.)
- *Resolution*: How precise does the data need to be?
- *Scope*: How much of the range of possible values needs to be conveyed?

For ease of reference, we refer to these abstract QoS parameters as the FRIARS values associated with information needs and with the way in which information is delivered. We note also that our candidate FRIARS parameters are related to, but expand and refine, the triad of abstract QoS parameters that Tom Lawrence of Rome Laboratory has advocated: Timeliness (refined as freshness and initiation time), Precision (refined as scope and resolution) and Accuracy (refined as accuracy and reliability above).

FRIARS values are specific to an information profile within a task. Thus, for example, a targeting task will require greater freshness on the enemy position data than will a task that assesses enemy force capability within an area. Thus, FRIARS values will represent the QoS requirements on the information needs for that task. In addition, we will work with allocation algorithm developers to operationalize the FRIARS parameters for the information control state which the allocation algorithm determines. This equivalencing of needs and capabilities definitions is required to compute "match" metrics which will be critical in performing optimization with regards to the commander's information policy—and in reporting to the commander how well his stated policy is achieving his intent.

In addition to these FRIARS values, we will encode a weighting value showing the importance of that QoS dimension for that type of information for that task—that is, Weight X FRIAR value X Information Profile X Task (where the "x" denotes "cross", not the multiplication operation). These weightings show the relative importance of 'getting it right' for each of these dimensions—that is, how much latitude the allocation algorithm should allow itself in meeting that information profile exactly as it has been stated. For example, while all tasks may have an initiation time value, some tasks may be very resilient with regards to deviations from that initiation time, while others may fail completely if they do not get their information exactly when

stipulated. We anticipate that the allocation algorithm will use these weightings to make tradeoffs in formulating an information control state.

This is not the only form of importance weighting that we will include in the information needs representation. We will also capture an importance weighting for the information profile in the task which it serves (Importance x Profile x Task) and a priority or importance value for the task itself in the commander's overall mission plan (Importance x Task). While these various levels of importance may be integrated into a single value for the allocation algorithm, it is important to keep them separate at the policy management layer to track their individual contributions for coordination with the commander's intentions and goals.

6.2.1.3 A Task-based Tool for Creating Mission-based Information Plans

Both the task representation and its linked information needs representation will be accessible to the commander via a task-based tool for creating mission plans. We have produced an early prototype of this tool for the year 1 effort. We envision the eventual tool as consisting of four primary components: A graphical user interface through which the commander accesses and manipulates task templates to formulate, review or edit a plan, a Task Template library which houses the known tasks and their associated information profile links, a Process Development Workspace representing a database linked to the GUI in which the commander does his building and editing of mission plans, and an Active Process Model Space (APMS) where the 'current' commander's mission plan lives when he is not working on it. Each of these components will be described in more detail below.

6.2.1.3.1 Graphical User Interface

Our recent work in using task model representations to develop control plans for Unmanned Air Vehicles (UAVs) [13, 14], and the ability to develop supporting, graphical model construction and editing tools using Honeywell's Domain Modeling Environment (DoME) tool, will enable us to rapidly produce a Graphical User Interface (GUI) suitable to the task representation and manipulation components' requirements. This interface will allow a commander to provide information policies, as well as information profile requests, as a byproduct of supporting his ongoing planning needs.

We have constructed a GUI of this sort, permitting direct viewing and manipulation of the task model itself, to demonstrate the basic utility of the IPSO-FACTO approach. One Playbook GUI is presented in REF _Ref482064299 \h Figure 18. Note that other interfaces are possible (e.g., a map or timeline tool) and would be pursued if user needs dictate.

To enhance the GUI, we would refine a graphical formalism created during the UAV effort for representing process models and provide links between it and the Task Template Library and APMS via Honeywell's powerful DoME toolkit, used as the basis of the year 1 prototype. The commander will work with the GUI (which itself must interface to the Template Library) to develop or revise a mission plan in the Process Development Workspace. The resulting model will be asserted into the APMS, where it can be accessed and reviewed to provide guidance to the allocation algorithm.

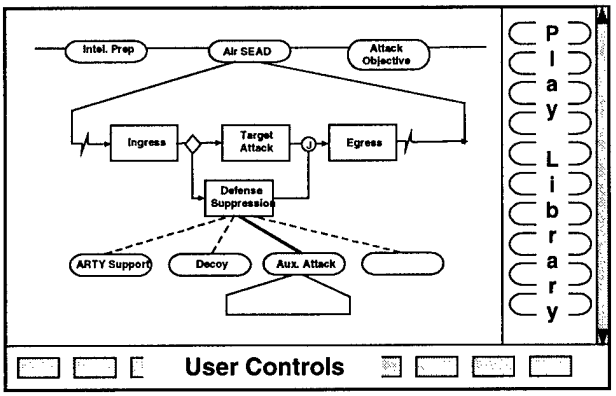

Figure 18. One Possible Instantiation of a Task Plan Capture Tool GUI

6.2.1.3.2 Task Template Library

Using the Task, Information Requirement and QoS representations described above, we will maintain a Task Template Library. This library will include both templates and larger, composed task and process structures. Users will draw components from this library and build or modify them into a process model for current use.

Every fielded IPSO-FACTO system will be equipped with a task template library. Initially the library will be just a "seed library" of domain-relevant task templates. However, as the system is used, the library will grow as users add specialized templates or partial models to the library for subsequent reuse. We will develop an indexing scheme to this library based on the role of the user creating the task, the recency of creation and on the hierarchical composition of the task domain which, in our experience with task model construction, is a general feature of most task domains.

6.2.1.3.3 Active Process Model Space (APMS)

When the commander has constructed a mission plan and wants to test its execution, he asserts it into the APMS. The APMS is where all concurrent collaborators in a given mission activity will access, maintain and manipulate their portions of the shared mission model. As such, it must be a shared data structure with inputs and change management capabilities accessible to all participants. A minimal implementation of the APMS is intended for initial phases of future work, sufficient for a single commander's use.

6.2.1.3.4 Process Development Workspace

The Process Development Workspace is, then, one or more "region(s)" of the APMS where the commander can develop new mission plan segments prior to actually inserting them into the mission plan. As such, the Process Development Workspace provides access to the current APMS to capture a "copy" of the current model that then can be modified in various ways prior to asserting the changes back into the mission plan. Additionally, this workspace allows the commander to test different mission plans and associated information control policies in order to determine which has the most desired outcome. For example, the commander could vary the priority of the tasks to be performed or even vary the tasks to be performed—what would be the

outcome if Task X was performed instead of Task Y, or what would happen if Task X were lowered in priority.

Note that the allocation algorithm can only give the commander its "best guess" at the outcome of his mission plan based on the commander's own guesses as to the likelihood of events occurring. Battlefield planning often consists of guessing games in which different scenarios are played in the commander's mind. The ability to test hypothetical scenarios optimizes this guessing game by incorporating known and predicted values for task variables, which might otherwise be greater than what the human's cognitive capabilities could effectively manage.

6.3 Creating Task Templates

The availability of a library of task templates suitable to feed the task-based policy specification described above is a non-trivial challenge. While it was not a focus of our Year 1 work, one of our subcontractors during the AICE program, Aptima, Inc., devoted effort to the development of an approach to building task templates and then to instantiating them for a specific battle plan. Their suggestions were delivered in a report which has been incorporated in this section.

6.3.1 Description of the Problem

Effective mission performance across the range of military operations is dependent upon appropriate information exchange within and between all organizational levels. It is a basic concept of the Agile Information Control Environment (AICE) that information exchanges should be driven by dynamic operational requirements and information exchange policies that are explicitly and implicitly directed by the commander. Further, implementation of the commander's information policy spawns new subsets of information policies throughout the organization to enable each echelon to accomplish its mission requirements. This results in organizational information policies at multiple levels: policies specific to the commander's intent and supporting policies that facilitate meeting commander's intent at subordinate levels. With current capabilities, translating the commander's mission concepts into enabling information policies and disseminating and implementing them throughout the organization would be an ad hoc and time consuming process. A tool is needed to rapidly translate mission requirements into enabling information policy, and to implement that information policy at all operational echelons.

6.3.2 Description of the Information Policy Management (IPM) Layer

The purpose of the Information Policy Management (IPM) Layer of DARPA's Agile Information Control Environment (AICE) is to capture and represent commanders' policies for information exchange in a specified mission environment, including the use of their communication and information resources. Essentially, the IPM layer translates commanders' concepts about how they wish to conduct a mission into parameters that are used to control the flow of information over the networks that support the mission. The commander develops a mission plan, which is used to define an information policy for the mission. This information policy is then used to determine the importance of various information exchanges. The current design for the IPM layer of AICE calls for the creation of "policy elements" that define the importance of classes of requests for information. A matching process searches the commander's policy (set of policy

elements), finds the policy element that best matches the source, destination, and exchange characteristics (information content) of the information exchange, and assigns an importance to the exchange request based on the importance specified in the matching policy element. Figure 19 provides a simple illustration of this concept. The importance of a request determines the resources that will be allocated to transmit the information and to generate and transmit responses to it.

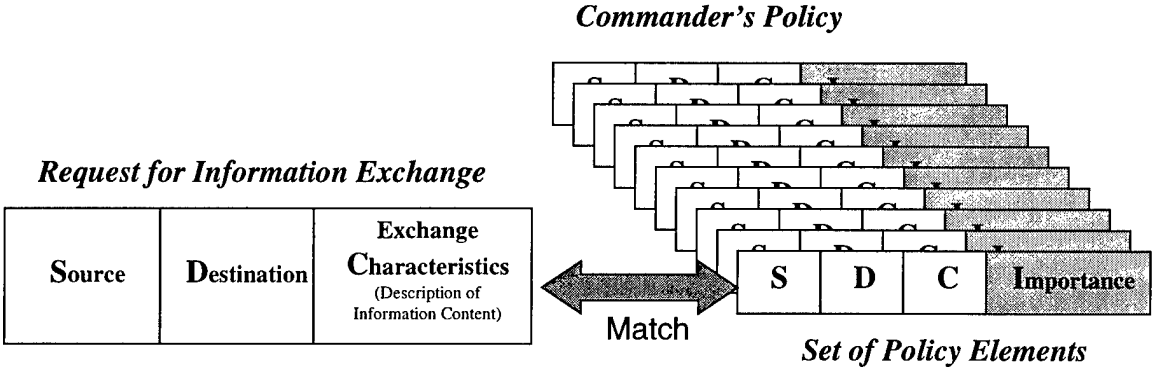

Figure 19. Determining the Importance of a Request for Information Exchange by Matching Against a Set of Policy Elements

6.3.3 Issues in Defining Information Policy Elements

We see several major issues in developing the policy elements as illustrated in Figure 19. First is *determination, adaptation, and utilization of the best typology for sources, destinations, and exchange characteristics (information content).* To create policy elements we will need to specify types of sources, types of destinations, and types of information content. The sources and destinations for each exchange must relate back to the network for which AICE is managing the information flow. An important issue will be level of granularity with which sources/destinations will be specified. This influences the level of granularity at which task templates will be developed. The typology for exchange characteristics (C) must add value in determining the importance of the exchange, i.e., the value of the C parameter is that it adds information (over and above the source and destination parameters) that improves the ability of IPM to set the importance of the exchange.

The second major issue is the *method for assigning importance to S-D-C classes* to create a policy element. The importance of the type of exchange must reflect the importance of that exchange for the mission task and should also reflect the utility of the information at each operational level. One method is to assign an importance directly to mission tasks and have all of the policy elements associated with that task inherit the importance of the task, assuming policy elements are completely contained within tasks. Another method is to determine both intra- and inter-task relationships to determine implications for assessment of task importance. This requires a fine-grained and sensitive method for assigning importance to S-D-C classes. The method should also take into account mission cycle, timing, timeliness requirements for the information exchange, the role or function of the user, and changes in the destination of information as tasks are redistributed during the mission.

The third issue concerns the *implementation of the information policy* throughout the organization. The commander's information policy will be broadly stated and reflect an operational perspective. To successfully implement the broad policy, each of the supporting organizational elements must develop supporting information policies appropriate to their perspective since each organizational element will have a different prioritization scheme based on its role and specifically assigned tasks. As the commander's operational intent is translated into supporting policies, a hierarchy of information policies results. A means to determine and filter information exchanges is necessary to ensure that each organizational element receives and appropriately processes its required information in a way that reflects its relevance to that element.

The complex nature of information policies and mission tasks highlights the importance of a fourth issue: creating simple, practical, *human-centered tools for developing information policies* and implementing them. It is not realistic to ask the commander to think about the mission in terms of connections between nodes in a network, and to directly specify a policy that is made up of S-D-C-I policy elements. Neither does it seem reasonable for the commander to explicitly develop supporting information policies at each organizational level. The IPM concept calls for the commander to develop a mission plan at a high level that specifies the tasks to be accomplished in the mission, and the creation (through a process yet to be defined) of a set of policy elements based on those tasks. These policy elements will be developed in greater detail as subordinate organizational elements further translate mission tasks and the commander's information policy into supporting information exchange requirements appropriate to their level and function. We address this issue in the next section.

6.3.4 Information Usage Activities

Figure 20 illustrates our vision for a task-based IPM tool. As discussed above, in our vision, a commander's IPM tool would permit, but would rarely require, the direct input of policy statements. Instead, commanders' information policies would be derived from each commander's battle plan. This vision is made possible by allowing the commander to quickly and easily assemble a battle plan by selecting and instantiating templates from an available library of appropriate *task templates*—each of which contains associated information needs with relative, within-task priorities. The commander instantiates the templates with data and parameters specific to the current situation, prioritizes and sequences them to create his battle plan. Once instantiated, the template becomes an *instantiated task* (or simply, a *task*) for the current battle plan. Since each instantiated task in the now-assembled plan contains associated information needs, the derivation of an information policy for that commander is a relatively straightforward process of collecting and generalizing the information needs associated with each task, sequencing them and applying cross-task dependency logic, and then weighting them by the priority the commander(s) assigned to the task.

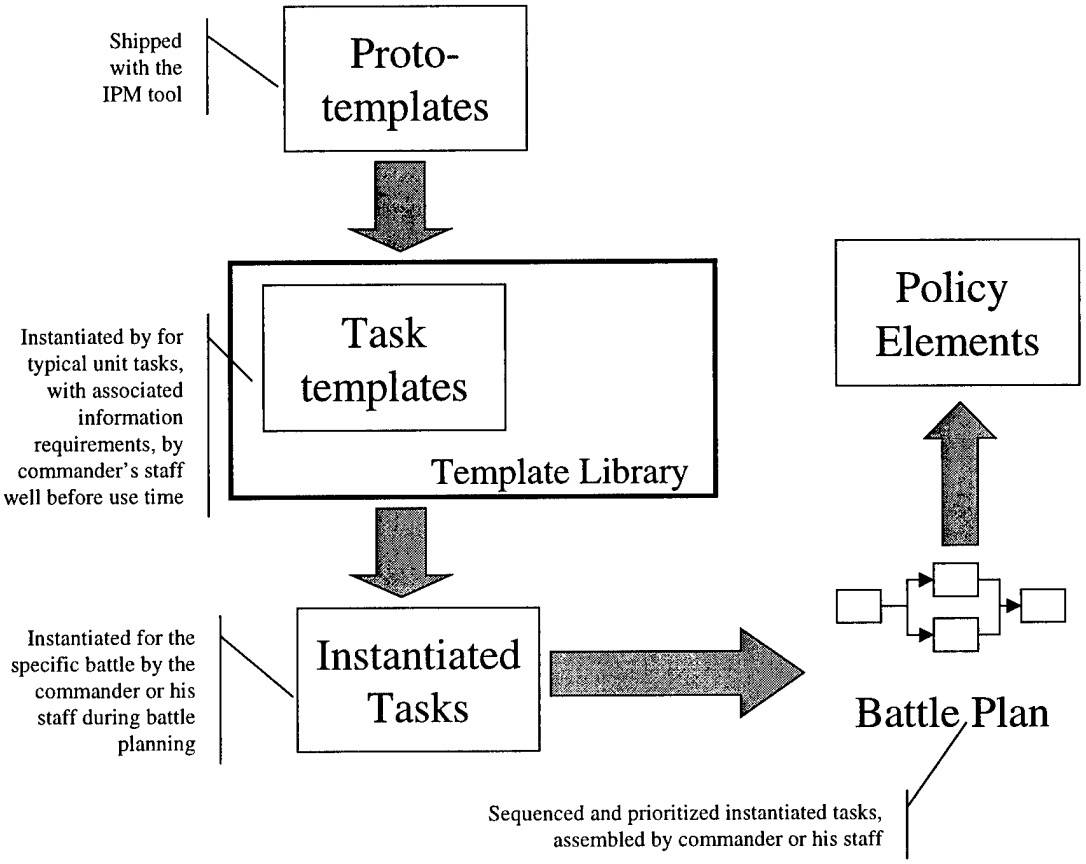

Figure 20. Vision for a Task-Based IPM Tool

Of course, this model simply begs the question of where task templates come from in the first place. Our belief is that it will be possible and useful to create a set of *'proto-templates'* to be shipped with each version of the IPM software. With minimal help, then, a local member of a unit's staff could use these templates, perhaps in conjunction with an advising 'wizard,' to construct unit-specific *task templates* as a part of the commissioning of the tool for use by that unit. Proto-templates would be very generic task types such as pre-planning, attack, reconnaissance, logistics support, movement, etc. Each of these has a minimal structure (e.g., attack generally requires a location, a target, and one or more weapon types to be specified) and at least classes of information types (e.g., attack requires information on enemy location and defense capabilities, with frequent updates). Most units will employ some version (perhaps many different versions) of most of these proto-task types, but will need to specialize them to create the library of task templates specific to their activities. As a simple example, flight units, sea units and ground units all engage in attacks, but the nature of the task and its associated information needs will differ greatly depending on which of these units is involved. Our envisioned approach would put the generic proto-task template 'Attack' into each of their hands and allow them to develop a library of task templates for their common operational activities long before taking the tool to battle.

A technique that may make this approach feasible is the identification of a set of "information usage activities" that are common across all tasks and that can be used as a guide to help build

the information needs associated with each template. Prior work by Aptima has identified seven underlying information usage activities, one or more of which is probably involved in every mission task — filter, dynamic integration, static integration, transform, store, seek, and disseminate. These activities (defined below) provide a structure for at least the construction of task templates and a guide to their instantiation for use in a specific battle plan. They may also, ultimately, prove to have utility for the creation of proto-templates and the transition from them to task templates. Specifically, they enable us to define questions that must be answered at each echelon to define the information policies and supporting policies. These questions may serve as the basis for an interface, prompting users to provide data that is not available in existing task templates. The information usage activities and their associated questions enable the system to ascertain how information is acquired, how it is used, and how it is disseminated in each task. They help us to learn the source, destination, characteristics, and importance of information exchanges for the task from the users' perspective. They also provide a structure that ensures that we capture all of the important information exchanges in the set of policy elements developed for each task both vertically and horizontally across all organizational echelons.

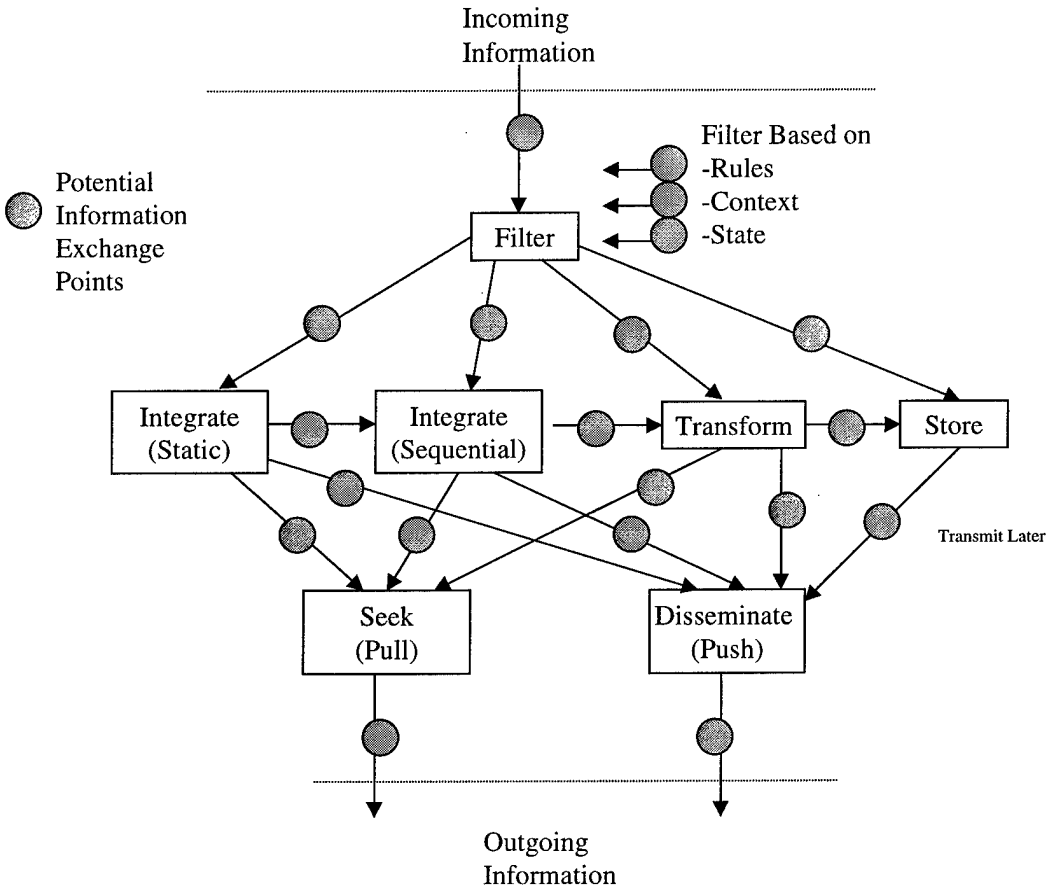

Figure 21. Possible Information Exchanges for a Task

The seven information usage activities constitute a model of information management and usage, as illustrated in Figure 21. This model has similarities to, but expands on, Wolf, Klein, and Serfaty's Information Management Model [15]. We suggest that these information usage

activities are present, at varying levels of detail and importance, in many, if not all, military mission tasks[9].

Information Filtering

Not all information received is of equal usefulness or importance; with the assessment of utility dependent upon the user/receiver/sender's role in the organization. Filtering is the process of deciding what to do with incoming information when some of that information is irrelevant, i.e., when information available is greater than information required. Filtering of information can be done in a number of different ways, including:

- Source/Time-based filtering. Information may be filtered based on source and time, e.g., discarding all messages older then 24 hours or discarding all messages from a source known to be unreliable. This is a relatively simple type of filtering.
- Destination-based filtering. Information may be filtered out based on the state of the entity receiving the information, e.g., the person to whom the message is directed may be in a state of high workload and unable to attend to the information, or the computer receiving the message may have gone down.
- Context-based (situational) filtering. Certain information may be filtered out as irrelevant because it is unrelated to current plans or goals. This is a complex filtering method requiring extensive information and understanding in order to apply the filter.

Filtering can involve many different types of information exchange. Most obviously, information is sent on to other activities (integration, transformation, storage) after it passes through a filter, and these exchanges may or may not involve sending messages over a communication network relevant to AICE. Dependent upon the filtering rules applied, alternative candidates to receive information may be identified. Less obviously, information exchanges may be needed to support the filtering process. For example, information on current sensor reliability may be needed in order to filter out data from unreliable sensors, and this information may or may not be transported over a network managed by AICE.

Information Integration (Static)

The integration or fusion of information from multiple sources is a fundamental activity of command and control systems. This integration may be done automatically by data-fusion algorithms or done by humans examining incoming information. Static integration is done in batch, at one point in time. However, it can feed a sequential information process. For example, data from multiple sensors may be fused into an integrated picture of the situation, and a

---

[9] Information may be exchanged at many different points, both within and between tasks, and these exchanges may or may not involve the communication network that is being managed by AICE. There may be multiple sources and destinations, different types of information content, and different importance levels for each of these exchanges, i.e., multiple policy elements that apply to each of the exchange points shown in Figure 21. Obviously, only those exchanges that occur on the AICE-controlled network are of interest for setting information policy.

sequence of such pictures may be integrated to form an impression about enemy intent based on movement patterns.

Information Integration (Sequential, Dynamic)

Dynamic information integration is simply the fusion of information over time. It may be performed sequentially, if reports are received at fixed intervals, or almost continuously if reports come in at unpredictable intervals. Dynamic integration is triggered when information is received after filtering or static integration, and it in turn triggers transformation, storage, dissemination, and seeking. Information integration may lead to the seeking of information if, for example, the integration reveals that important information is missing.

Information Transformation

Information transformation is the "use" or "consumption" of information to make decisions or take actions. Transformation adds value to the information rather than simply transmitting it. The output of transformation is greater than the input, i.e., new information is produced. For example, a commander might view information on enemy location, make inferences about enemy intent, and issue commands based on those inferences. This process draws on the information in the commander's head (experience and knowledge), but, from the point of view of an AICE information-exchange network, the commander has produced new information. There are information transfers both into and out of the transformation activity. Transformation may lead to information seeking (requests for information), dissemination of information (e.g., commands), or storage (perhaps for later dissemination).

Information Queuing and Storage

Not all information is transmitted or acted on immediately. Information may be stored for later use as a result of filtering or as a result of transformation. Sending information for storage is a type of information exchange (perhaps of less importance than other types of information exchange) that may or may not be accomplished via an AICE network. Information from storage may also be disseminated at a later time, and that also may or may not involve an AICE network.

Information Seeking

Searching for new information—information "pull"—is a fundamental activity of command and control. Seeking occurs when information available is less than information required to make a decision with an acceptable likelihood of being accurate. Associated with information seeking may be the need to adjust or instruct information-gathering devices/nodes. So, for example, an information-seeking exchange could be a command to a radar system to send a report on a certain area using certain search parameters. It could also be a request from HQ to an intelligence unit to send updated reports on enemy activity.

Information Dissemination

Information dissemination is the process of sending relevant information to other nodes or tasks. For example, an Air Tasking Order may be simultaneously sent to multiple locations. This activity, which almost certainly involves the AICE network could be simple (where few

Source/Destination communication links cross between tasks) or more complex (where multiple Source/Destination communication links cross between tasks).

We recognize that information usage activities reflect tactical and operational perspectives appropriate to the organizational level. It seems rare that a task or node will lack at least one input source and output destination in order to enable its integration with other tasks across the battle plan. It seems more likely that there will be many source/destination links that go across tasks. A general example and a specific example in the form of a conceptual Case Study for a Deep Attack task are provided in subsequent sections.

6.3.5 Construction of a Task Template Library

The performance of a task typically requires many different types of information exchange, each with associated policy elements. This raises the question of how best to define these exchanges in order to build a task template library and database.

We suggest that the information usage and management activities described above provide a natural structure for eliciting the information that is needed to populate task templates with mission-specific data—that is, for making the transition between the *task template* and *instantiated task* boxes in Figure 20. Specifically, we will use information activities to guide the development of questions that a hypothetical template wizard tool could use to elicit the mission data required to construct an information policy. For example, to understand Information Seeking activities, the wizard must step a user through addressing such questions as "What information is sought concerning this task?" "What is the priority of the required information?" and "Under what conditions is information sought in this task?" To understand Transformation activity, the wizard must step the user through answering such questions as "What decisions are made in this task?" "What heuristics or algorithms are used?" and "At what point is the information sufficient to perform transformation?" These questions will be assembled in a table such that each user will get only the questions that s/he requires. The organization of the template will be standardized to maximize usability.

Table 4 shows the elements of a task template table. A richer set of representative questions is presented in Table 5. How might these questions be used? The wizard would query the commander or staff to assist in implementing an operational plan (OPLAN) using a standardized task list. It would then present relevant questions and default answers or mission-specific answers, if available, for the mission tasks. The commander's staff would review and refine this material. Table 5 shows a notional template and illustrates the type of questions to be asked. (It is not intended as an interface design, however.)

Table 4. Elements of a Task Template Table.

Activity	Example Questions/Answers	Information Exchange		Typical Information Content	Relative Importance for task	Policy Elements for task			
		Source	Destination			S	D	C	I
A	*Representative questions* B								
	C	D	E	F	G	H	I	J	K

Table 5. Building Task Templates: Examples of Questions Generated from Information Usage and Management Activities for Task X

Task X

Activity	Example Questions/Answers	Information Exchange		Typical Information Content	Relative Importance for task	Policy Elements for task			
		Source	Destination			S	D	C	I
Seek	What information is sought concerning this task? What is the priority of information requirements? Under what conditions is information sought in this task? How (at what points in time, in response to what events) does this task involve asking questions or getting more information? What supporting information is required? Are there multiple requirements for this information between tasks?								
	Information sought for Task X	from whom	to whom	Typical content	How important is this information for Task X?				
Filter	How is this task and its elements distributed throughout the organization? Is all of the information that comes in useful for this task? How is useful information identified? What do you need to know to decide if information is useful? Who will require this information? Are there criteria for sufficient information to allow decisionmaking? What are the priorities associated with the information for all intended users? Are there time latencies that will impact the value of the information?								
	What information could be used to filter what's important from what's not?	from whom	to whom	Typical content	How important is this filter for Task X?				
Integrate (static)	How is this task related to other tasks that might require all or some of this information? What are the multiple sources of information for this task? Is all information needed for decision making received simultaneously? Is the information a result of raw or combined data? How is the data combined? What is done with the information after it is combined? What length of time can be considered static, e.g., seconds, minutes? Is there a means to assess validity of the information? Is there a prioritization to the information? How are conflicts between information resolved, e.g., multiple sensor inputs?								
	Where might you gather information from multiple sources that must be fused?	from whom	to whom	Typical content	How important is this information for Task X?				
Integrate (dynamic)	Is information needed for decision making received over time (rather than simultaneously)? How frequently is new information received? What is done with the new information? How is this task related to other tasks that might require all or some of this information? Is there criteria for sufficient information to allow decisionmaking? What are the priorities associated with the information for all intended users? Is the information a result of raw or combined data? How is the data combined? What is done with the information after it is combined? Is there a means to assess validity of the information? Is there a prioritization to the information? How are conflicts between information resolved, e.g., multiple sensor inputs?								

	Where might you gather information that changes/develops over time?	from whom	to whom	Typical content	How important is this information for Task X?				
Transform	What decisions are made in this task? What information is used to make them? What algorithms are used? What knowledge sources are tapped? Who is informed? How are outputs of this task related to inputs to other tasks? Is there an iterative cycle for information development within this task? At what point is information developed in this task sufficient to make required decisions? How are information elements prioritized?								
	What decisions are made in Task X?	from whom	to whom	Typical content	How important is this transformation for Task X?				
Store	Does this task involve storing data that are not passed on immediately? How is this data aggregated to develop required information? How are some information elements retained as data for aggregation as well as processed for support to other tasks? Where are the data and information elements stored? Under what conditions are the data and information elements released from storage? Is some information retained as well as processed for dissemination simultaneously? Are storage limitations identified, e.g., time, type, size? What are the triggers to pull data from storage?								
	What information might be received during Task X that should be stored for later?	from whom	to whom	Typical content	How important is this filter for Task X?				
Dissemin-ate	Is information from this task disseminated? What messages/information types are sent as a result of this task? To whom are messages information sent? How is this task related to other tasks? Does this information flow vertically, horizontally, both? Is dissemination of this information time sensitive? How does the priority of this information vary between elements of the organization?								
	What information has to be passed on to others at the end of Task X?	from whom	to whom	Typical content	How important is this filter for Task X?				

The template for each task is decomposed into the seven information usage activities (i.e., seek, filter, integrate (static), integrate (dynamic), transform, store, and disseminate) in the column on the left (A). Within each activity, we list a broad sample of the questions that might be asked (B). The answers to these questions will provide the information exchange requirements to complete that activity in support of the specific task. (Only a sample of questions is provided in this example.) These questions may be asked at all or some of the organizational levels, but would be customized to the specific operational echelon that is completing the template. The questions would be individually presented (C) to elicit anticipated information sources and destinations. The source (D) and destination (E) of the information would be specified. Each information exchange would further indicate typical information content (F) and an assessment of importance of the information exchange (G) relative to completing that information usage activity in support of the specified task. These constitute the policy elements discussed earlier. Where an information element has utility for more than one operational unit, the source (H) of the information element for each operational unit (I) is specified (using a numeric code, in the case examples that follow). This supports increasing focus and granularity at sub-echelons and provides for identification and assessment of triggers based on changes in mission and importance cycles. Information contents (J) and the importance of the information (K) to each destination unit are also specified. Initial values are provided for these Policy Elements by SMEs, but the values can be adjusted if the situation warrants.

A more detailed template is provided in the following Conceptual Case Study section. As the user, in conjunction with the wizard, begins to construct information policies, it would elicit more detailed data from subordinate echelons. Two characteristics of this interaction are important from a human engineering standpoint. First, the questions will be formulated in terms that are meaningful to users at each operational level. Second, only those questions necessary to "fill in gaps" will be presented to the user at each echelon.

As policy elements are built up for tasks between missions over time, we anticipate considerable commonality among similar tasks in the template library, allowing the wizard and users (depending on level of automation), to cut and paste specific elements, with some modification, between similar tasks. For example, the policy elements associated with information integration in one task may be very similar to those for information integration in another task.

The task templates would be developed working with subject matter experts. Additional existing planning tools, doctrinal publications, tactical publications, field manuals, and other documentation would be reviewed and assessed for relevance and appropriate materials to develop task templates extracted, e.g., existing task matrices, operational/mission diagrams, organizational charts.

One interesting characteristic of experts in many fields [16, 17] is a reliance on multiple representations of a problem and rapid shifts between those representations. We propose that this system provide a variety of representations. We have mentioned the textual representation that includes questions and answers. In addition, users may benefit from diagrams of task flow such as that shown in Figure 22; diagrams of information flow between tasks; and diagrams of information flow between operational units. These representations would support both review of task and information flow, as well as revision.

6.3.6 A Conceptual Case Study Example: JSTARS Deep Strike Targeting Scenario

To illustrate the concept of use of a template instantiation wizard as described above, we provide a very limited conceptual scenario below. Templates for two representative tasks (from the task flow diagram in Figure 22) are provided. The first task relates to preplanning and was selected to demonstrate a query process relevant to a moderately complex information exchange environment with information resources that extend from the tactical to strategic level. The second task, monitoring for targets, represents a very limited information exchange environment in order to demonstrate a more focused query environment. These task templates represent an aggregation of questions that would be disseminated throughout the organization based on the need to develop increasing levels of detail.

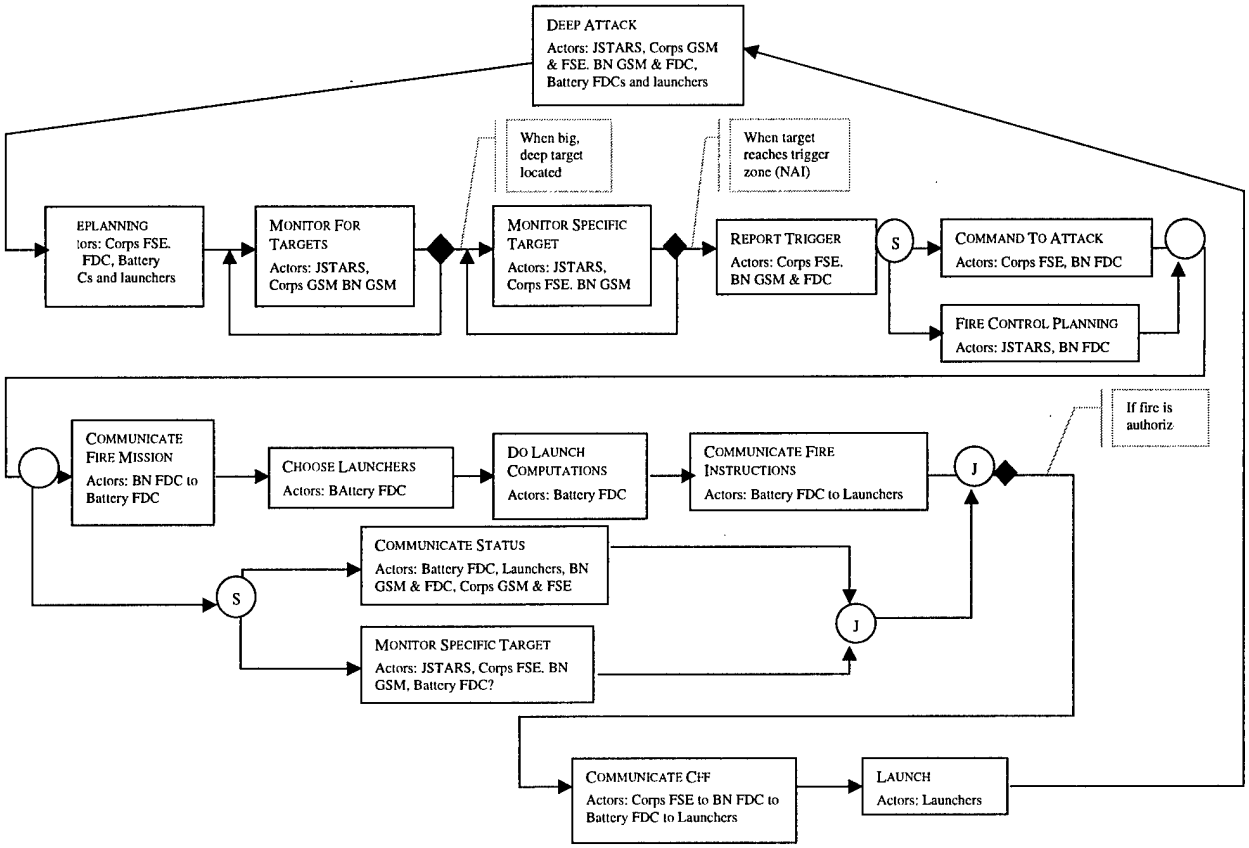

Figure 22. Task Decomposition of Deep Strike Scenario

Following is a brief background of the scenario as well as a task sequence.

Scenario Background:

A peace enforcement operation in northern Africa has been on-going for a week. The conflict represents an escalation from a preceding month-long peacekeeping operation. Country X and Country Y have been engaged in a border dispute for several years over recently discovered energy reserves. The dispute has turned violent, resulting in multiple border clashes between ground units patrolling in the disputed area. These clashes have escalated in intensity and have the potential to destabilize the region if a war should result. The US, acting with UN authority, has moved to intervene to maintain the peace using elements of an Army Corps previously stationed in Country Y. A joint task force, TANGO, has been established and assigned the mission to prevent escalation of the conflict.

US ground forces are still in the process of building up. Initial US ground forces in place consist of Corps Headquarters, one mechanized division and two artillery battalions (with associated firing batteries) previously stationed in nation Y as well as USAF Joint Surveillance, Targeting and Reconnaissance System (JSTARS)[10] assets based in country Z, 500 miles away. Currently, the host nation, Y, has not authorized use of air facilities or positioning of naval assets in national waters

10 JSTARS is a scanning radar for detection of moving targets on the ground, and is carried on a U.S. Air Force E-8A aircraft platform.

other than for direct support of the buildup of the US ground forces. Intelligence, indications and warnings show that Country X intends to mass forces including two armored and mechanized divisions along its border with County Y in preparation for an invasion to seize the disputed territory. The disputed area is mountainous and consists of heavily covered transit routes. Detection of movements of ground units is anticipated to be achievable in limited areas and at limited intervals due to the terrain and adverse environmental conditions.

US forces have been positioned to detect and counter movements of hostile forces into the disputed area resulting in multiple Named and Targeted Areas of Interest (NAI/TAI). However, US forces have not been allowed to enter the disputed area, which is located within the recognized borders of Country Y. The NAI and TAI have variable and varying expectations of detection windows for detection and engagement of moving forces based on changes in environmental conditions and were established to reflect likely buildup and border crossing areas. Surveillance assets to cover the disputed area include intermittent use of national satellite assets and the JSTARS. However, due to other higher priority contingencies, use of national assets is expected to be minimal. Country X has been warned not to cross its border into the disputed area within Country Y. In order to counter movement of hostile forces into the disputed area, coordinated deep strikes between the Army ground units and the JSTARS will be required.

The JSTARS platform flies in an orbit at a safe standoff distance from the battle and collects and processes radar imagery in support of USAF, multi-service, and joint operations. For Air Force missions, the imagery is analyzed and targets are developed by analysts on board the E-8A. For Army targeting and intelligence missions, the imagery is downlinked to ground processing stations located with various Army command posts on the ground. In this scenario, the JSTARS is assumed to have an on-station time of approximately 8 hours and a transit to/from the patrol area of one hour each way. Tanker assets are not available. Continuous JSTARS support is assumed with a turnover between aircraft entering/leaving station. Limited/intermittent use of unmanned aerial vehicles and intelligence reports from within the disputed area are available. In this limited scenario, Deep Attack on moving convoys by US Army forces is supported by JSTARS through an interface with the Corps Support Weapon System (CSWS). This requires extensive communication between the JSTARS E-8A, the Army Corps G2 (Intelligence), the JSTARS Ground Station (GSM), the CSWS Fire Direction Centers (FDCs), and the CSWS Battery itself (i.e., "the shooter"). This JSTARS Targeting Scenario can be broken down into 14 tasks (see Figure 22): preplanning, monitor for targets, monitor specific target, report trigger, command to attack, fire control planning, communicate status, monitor specific target (2), communicate fire mission, choose launchers, do launch computations, communicate fire instructions, communicate Calls For Fire, and launch. We will use two of these tasks, Preplanning and Monitor for Targets, to provide examples for development of the task template.

Task Descriptions:

Task 1: Preplanning. Preplanning is an extensive effort that includes all aspects of preparation for deploying forces and conduct of operations in a specified operational environment. For purposes of this scenario, the preplanning of interest is focused on stopping the movement of an armored column into the disputed area via a coordinated deep strike/attack. To demonstrate how task importance varies across operational echelons, preplanning can be looked at from the commander's broad operational perspective of what are the implications of the convoy's movements with respect

to accomplishing his/her mission to the artillery battery's perspective of where is the target and what it should shoot it with. In this case, information such as location of the target would have different importance relative to each echelon. The commander would take into account a variety of concerns such as enemy intent, political implications of collateral damage, potential for escalation, best time/place to target, etc. The artillery battery is much more focused on target type (hard/soft), target disposition (concentrated or distributed), available munitions/weapons, and expected location at time of fire. Specific queries would be limited to those questions, and phrasings of those questions, necessary to elicit the information at each echelon to develop the necessary inputs to implement an appropriate information policy.

Table 6 provides an example of a task matrix that would provide the elements from which a coherent, aggregated set of information policies could be developed for the Preplanning Task. The table specifies questions to be asked for each type of information usage activity in developing the task template, and gives examples of the sources, destinations, contents, and importance of the information exchanges that were identified in response to those questions. Note that Sources and Destinations are specified using a numerical code defined at the bottom of Table 6, and that importance is rated on a scale from 1 (not at all important) to 5 (very important).

Extensive preplanning and control is required in order to conduct Deep Attack on moving targets, including terrain analysis during operational/mission planning. In this scenario, the Fire Support Element (FSE) at Corps identifies the likely enemy avenues of travel based on intelligence inputs and reconnaissance. Along each route, they identify a series of Named Areas of Interest and Targeted Areas of Interest. The Targeted Areas of Interest are preplanned fire zones placed in areas where convoys are expected to be detectable by JSTARS radar and that are conducive to attack by CSWS munitions. The Named Areas of Interest are simply coordination points preceding each Targeted Area of Interest. When a target enters one of these Named areas, final preparations for attack begin. This preparation includes predictions by the JSTARS GSM about estimated arrival time at the fire zone. In this scenario, the NAI are established on Country X's side of the border and are linked with TAIs in the disputed area. It is important to note that the tasks and supporting templates represented here are a very limited set of the specific activities that would be required in even a simple scenario. Some additional representative preplanning subtasks that would require task templates follow:

1. Perform Intelligence Preparation of the Battlefield
    a. Conduct Terrain Analysis
        i. Identify avenues of approach
        ii. Identify effects of current and predicted weather on trafficability
        iii. Identify areas mask/expose movement and attack intention
    b. Locate blue force positions and strengths
        i. Locate hostile garrison areas (from All-Source Intell)
        ii. Locate objectives (from All-source Intell)
        iii. Estimate hostile unit strength (from All-source Intell)
        iv. Estimate convoy size and mix for each unit/garrison
        v. Estimate rate of movement for each unit

vi. Identify avenues of approach and adjusted movement rates

c. Plan Sensor Flightpaths, Orbits, Periods of Activity Surveillance

i. Determine available sensor platforms, times of availability, and airbase locations
ii. Review incoming requests for sensor coverage and on-station time
iii. For each request, examine flight requirements.

1. Determine/plan orbit/flight path required to meet constraints of surveillance range and LOS terrain masking
2. Determine flight path to orbit point considering distance and fuel
3. Assess vulnerability to enemy air defense
4. Assess vulnerability to changing weather
5. Accept/modify/reject requests for coverage

2. Cue/Task Sensors

a. Communicate proposed surveillance flights/missions to G2 and FSE
b. Communicate mission taskings to airborne surveillance units

3. Execute Collection Assignments

a. Develop All-Source Intelligence Analysis (Enemy Situation Estimate)
b. This includes the same tasks as Intelligence preparation of the battlefield, only operating with dynamic information during the battle

4. Develop Deep Attack Plans

a. Review Intell Prep of the Battlefield analysis
b. Conduct terrain analysis
c. Identify chokepoints, areas of delay and diversion due to terrain
d. Identify sensor line of sight (LOS) masking at possible chokepoints
e. Identify terrain constraints on weapon delivery (i.e. trajectories versus position of targets and terrain obstacles)
f. Select areas along routes for targeting
g. Review high priority targets for deep attack (from the commander)
h. Prepare requests for sensor on-station time and area coverage
i. Modify plan based on available surveillance support

Task 2: Monitor for targets. In this scenario, while monitoring the Corps Area of Interest for significant convoy movement (a task as indicated in Figure 22), the JSTARS GSM at the Corps G2 cell detects and identifies a large armored convoy moving toward the disputed area. This Corps GSM sends a message about this target ("hands it off") to the GSM at the CSWS Battalion FDC. Table 7 provides an aggregated task template (i.e., a task template that contains questions that would be asked of different users, as appropriate for each user's level and responsibility) reflecting the specific requirements to develop an information policy to support meeting task requirements. Again, the queries would be presented to the degree and in a form appropriate to specific users in

the organization until a coherent information policy is developed. Inter-task dependencies would also be assessed.

Although preplanning and monitor for targets were the tasks chosen to illustrate the task templates for the JSTARS deep strike scenario, all of the tasks in the task flow diagram of Figure 22 would have separate task templates.

6.3.7 Summary

This report of work performed by Aptima, Inc. has provided a concept for a structured approach to creating task templates in a template library customized to each military unit and then, later, instantiating them with the specific information required for the tasks in a specific battle plan. This concept advocates use of a wizard which functions to populate task templates that are used to derive information policies to facilitate accomplishment of the specified tasks. The wizard elicits information from existing related task templates and databases to the extent feasible, and then steps human users through a knowledge acquisition template, based on a defined set of information exchange activities. A case study representing the application of the task template concept with respect to defined information usage activities was also presented for elements of a Deep Attack scenario. This case study demonstrated the spectrum of perspectives at each organizational level that must be supported by the task templates and wizard. The specific form of the wizard, i.e., the query environment, is envisioned to be tailorable to specific user preferences and will only require sufficient input to enable the wizard to complete the task templates.

Several issues will need to be resolved concerning how the wizard will function with respect to changes in mission and importance cycles. Additional work will be required to establish an appropriate typology for the various information policy elements. Further, inter- and intra-task relationships will need to be explored along with a mechanism for the wizard to accommodate them.

Table 6. Task Template for Preplanning Task

Task 1: Preplanning

Activity	Example Questions/Answers	Information Exchange: Source	Information Exchange: Destination	Typical Information Content	Relative Importance for task	Policy Elements for task: S[1]	D	C	I[2]
Seek	*Representative questions* **What information might be sought concerning this task?** What is/are likely points of detection? What is/are expected contents of convoy? What is/are Rules of Engagement (ROE)? What is/are mission priorities? What is/are threat order of battle? What is/are resources to engage? What is/are priorities of activities underway? What information resources are available? What is/are geographic area characteristics? What is/are current/anticipated weather conditions? What is/are terrain with respect to enemy movement? What is/are weapons limitations? What is potential for collateral damage? What is target composition (number of trucks/tanks/APCs, spacing)? What is/are deconfliction issues? What is/are coordination requirements with other friendly units? What is/are level of kill required? What is/are enemy ability to detect threats? What is/are known disposition of enemy forces? What is/are known disposition of friendly forces? What is/are known disposition of neutral forces? What is/are enemy intent?								
	What is/are likely points of detection?[3]	J5 or other planners, intel, operations people (who get spot reports), UAV controllers, tasking plan for the sensor (planning cell)	all command elements: jstars, corps, battalion, battery	geographic coordinates, times	Very Important (across the board)	3 8	4 5 6 7	geographic coordinates, times	5
	What is/are expected contents of convoy?	Intel	all command elements: jstars, corps, battalion, battery	classification of contents (e.g., war materiel vs. consumer goods)	varies, depends on destination	1	4 5 6 7	classification of contents	1 1 2 4
	What is/are ROE?	Command: CJTF	all command elements: jstars, corps, battalion, battery	criteria for engagement	Very Important (across the board)	2	4 5 6 7	criteria for engagement	5
Filter	*Representative questions* **What do you need to know to decide whether the information is useful?** What is the type of column (Armored? Soft?)? Composition of armored column? What is expected arrival time? What is expected speed of advance? What is expected target type composition? What is detectability given weather? What is detectability given sensor? What echelon needs what information when? What other tasks require/provide relevant information?								
	What is the type of column (armored? Infantry? Intermixed with neutrals?)	Battalion, Intel, Corps	all command elements: JSTARS, corps, battalion, battery	target type	varies, depends on destination	6 1 1 5	7 4 5 6	target type	5 3 4 2
	Composition of armored column	Battalion, Intel, Corps	all command elements: jstars, corps, battalion, battery	distribution, co-mingling (neutrals in the column?)	varies, depends on destination	6 1 1	7 4 5	distribution, co-mingling	5 4 4

						5	6		4
Integrate (static)	*Representative questions* **Where might you gather information from multiple sources that must be fused?** What are the available sensors? What are the available intelligence and operational including reporting frequency?								
	What are the available sensors, including intell?	Battalion G3, JTF Operations J3, Corps G3, J-2, NSA, Corps G2, Battalion G2	battery, Corps, Battalion, JSTARS	JSTARS (or more generally a sensor name and its parameters) Intelligence estimates including intended destination and composition	varies, depends on destination, speed of advance and compositio n	6 3 5	7 5 6	sensor name and parameters, time?	1 4 2
Integrate (dynamic)	*Representative questions* **Where might you gather information that changes/develops over time?** What are the available sensors? What are the relevant intelligence sources? What is the frequency of reporting, e.g., intelligence and situation reports? Are non-military resources available, e.g., Commercial media, non-government agencies, other government agencies? What is the criteria for sufficiency of information to support decisionmaking? What is the perceived accuracy/reliabilitiy/priority of sensors?								
	What are available sensors to dynamically monitor the situation?	Battalion G3, JTF Operations J3, Corps G3, J-2, NSA, Corps G2, Battalion G2	battery, Corps, Battalion, JSTARS	Sensor names/param eters, routine reporting cycles/times, OPSITS/SITR EPs,	varies, depends on destination, speed of advance and composition, assessed intent	6 3 5	4 5 6 7	sensor name and parameters, time?	4 4 3 1
Transform	*Representative questions* **What decisions are made on this task?** How can I use this information to conduct a terrain analysis? To determine the target list, NAI, and TAI? To determine available sensor platforms, times of availability, and airbase locations? To determine likely detection windows? To prioritize targets? To anticipate target times/schedule firing assets? To determine hostile intent? To determine intent of target movement/intended destination? Determine operational implications? Select types of munitions and numbers of rounds? Assess lethality requirements, e.g., mission kill vs. destruction?								
	Determine flight path for surveillance assets to orbit point considering distance and fuel	Airborne Surveillance Unit/Platform Commanders	Airborne Surveillance BN CMDR	information on distance and fuel	not very important		6	Distance; amount of fuel	2
	How can I use this information to conduct a terrain analysis?	G/J-2, G/J-3, FDC, FSE, JSTARS,	Corps, Battalion, Battery	Impassability, terrain types, chokepoints, covering points, hardened areas, type of cover	Varies: very important for planners and upper echelons, low importance to battery	4 5 6	5 6 7	NAI, TAI, chokepoints, estimates of munitions effectiveness, collateral damage potential?	4 3 2
	To determine intent of target movement/intended destination?	G/J-2, G/J-3, FDC, FSE, JSTARS	Corps, Battalion	Likely destination, speed of advance, composition, operational/t actical implications	Varies: very important at Corps and other operational level; moderate at Tactical/batta lion level; minimal importance at battery level	4	5 6	Operational/t actical implications, path of intended movement	5 4

						5 6	6 7		4 1
Store	*Representative questions* **What information might I get during preplanning that I should store for later?** What potential targets are in the area? Where are Launchers? Launcher ranges? Named Areas of Interest? Targeted Areas of Interest? Required type/number of munitions? Sensor availability? Munitions available/expended and type? Expected destination/progress points/times for column? Detection windows? Potential collateral damage? ROE?								
	Location of Launchers	Battery, battalion, corps	Battalion, corps, JSTARS	Geographic coordinates	Very important at battalion and Corps level; critical at battery level	7 6 5	6 5 4 3	Fields of fire, geographic location	5 4 4 4
Disseminate	*Representative questions* **What information has to be passed on to others at the end of preplanning?** What are the NAI and TAI? Expected target types, path of intended movement, composition? Requirements for surveillance and targeting? Weapons/munitions requirements? Prioritization of targets?								
	What is the NAI?	JTF, Corps, Battalion	JSTARS, Corps, Battalion, Battery	Geographic coordinates	Very important at all levels	3	4 5 6 7	Locations, anticipated targets/types	5

[1]codes for S (source) and D (destination): 1 =national intelligence; 2 = command (CJTF); 3 = JTF Operations; 4 = JSTARS; 5 = Corps-GSM; Corps-FSE; 6 = Battalion-GSM; Battalion-FDC; 7 = Battery; 8 = JTF Intelligence; 9 = National assets; 10 = non-military sources, e.g., NGOs/other agencies/Media

[2]Rating of importance on a scale of 1 (not at all important) to 5 (extremely important)

[3]a function of sensor, target, terrain, weather, dynamics of sensor tasking (e.g., is it in the area to observe targets)

Table 7. Task Template for Monitor for Targets Task

Task 2: Monitor for Targets

Activity	Example Questions/Answers	Information Exchange		Typical Information Content	Relative Importance for task	Policy Elements for task			
		Source	Destination			S[1]	D[1]	C	I[2]
Seek	*Representative questions* **What information might be sought concerning the task "monitor for targets"?** Is there movement near or into the NAI? How complicated is the tactical/operational picture? How many tracks are in the NAI? What tracks can be designated as targets? Is target a known hostile or assumed hostile? What is the targets' priority at Corps, Battalion, and Battery level? How fast is target moving? In what direction? What is the target's anticipated mobility and speed of advance into and through the NAI? What is target's intent? What is the reliability of my sensors? What sensors are available? What is the expected detection capability of my sensors? What are the counter-detection capabilities of the target? Are there additional non-hostile tracks in the vicinity? What intelligence and operational reports are available regarding estimated/known position of targets? What are the anticipated reporting frequencies? Has expected type/composition of target changed, e.g., split forces? What combination of information provides sufficient criteria to positively identify target? To designate the target? To make the attack decision? What are the ROE? What are the Critical Information Requirements as prioritized at Corps, Battalion, and Battery level? What are the CCIRs as prioritized at the JSTARS? What CCIRs relate to identify of this target and what assets are currently be employed to meet them? What additional/correlating sensors will be available in the next [pick timeframe of interest]? Given the terrain analysis, anticipated sensor availability/reliability, and anticipated target intent/destination; is there an optimum point to engage the target? What is the anticipated likelihood of losing track on the target?								
	What hostile, neutral, friendly units are in the area of interest?	Joint Stars	Corps GSM, Battalion GSM	information about the area	very important	4	5	graphics and RADAR imagery of the area	5
	What is target's intent?	Intel	Corps GSM	information about the target	very important	1	5	intelligence report	5
Filter	*Representative questions* **What do you need to know to decide whether the information is useful?** Are the forces near the NAI friendly, neutral, or hostile? Which organizational elements require information for contacts in specific NAI? What information is required by each organizational echelon regarding the contact, e.g., intent, path of movement, estimated position, composition, distribution? What is the priority of each NAI relative to organizational echelon? How old is the information? Is the information still required to support decisionmaking and at what echelon? Which sensors hold contact? What is their perceived reliability/viability at each echelon?								
	Are the forces near the NAI friendly or hostile?	Intel	Corps GSM	information about the forces	very important	1	5	Intelligence Report	5
Integrate (static)	*Representative questions* **What information might you gather from multiple sources that must be fused?** What is the current status, e.g., day/night/lighting, weather, visibility, of the area? What sensors have contact on tracks of interest? Are there positional differences between sensors? Are there significant potential time differences between detections? Are there disparities between sensors that result in conflicting reports?								
	What is the current status of the area?	Intel, Joint Stars	Corps GSM	information about the area	very important	1	5	intelligence report	5
						4	5	graphics and RADAR imagery of the area	5
Integrate (dynamic)	*Representative questions* **What information might you gather that changes/develops over time?** What units move into and out of the area? What is the current status, e.g., day/night/lighting, weather, visibility, of the area? What sensors have contact on tracks of interest? Are there positional differences between sensors? Are there significant potential time differences between detections? Are there disparities between sensors that result in conflicting reports? Is there stored information that is available to augment new information?								
	What units move into and out of the area?	Joint Stars	Corps GSM	information about the area	very important	4	5	graphics and RADAR imagery of the area	5

Transform	*Representative questions* **What decisions are made on this task?** Does identified target match Deep Attack Criteria? What is target identity and intent? What means are best suited to counter the track? Is the track a threat? At what point should the track be engaged? Are there expected tracks/movements in a given timeframe? Has a friendly/neutral/hostile track exhibited characteristics sufficient to alter its designation? Have changes in mission/importance cycle resulted in specified priorities of potential targets?								
	Confirm target identity and intent.	Corps FSE, Corps G2	Corps G2, Corps FSE		LOW	5	5		2
Store	*Representative questions* **What information might I get while monitoring for targets that I should store for later?** What is target composition (number of trucks, spacing)? What is potential trafficability at specified times? What are anticipated detection windows in given areas? What are the characteristics that would allow classification of a contact?								
	How many trucks comprise the target?	Intel	Corps GSM	distribution of target	medium	1	5	distribution of target	3
Disseminate	*Representative questions* **What information has to be passed on to others at the end of monitoring for targets?** Is target a possible threat? At what level? Who is responsible to track/counter movements in specified NAI and TAI? What is criteria to report track to each echelon?								
	Report potential target	Corps GSM	Corps FSE, BN GSM	likelihood that target is a threat	LOW	5	5 6		2

[1]codes for S (source) and D (destination) 4 = JSTARS; 5 = Corps-GSM; Corps-FSE; 6 = Battalion-GSM; Battalion-FDC; 7 = Battery

[2]Rating of importance on a scale of 1 (not at all important) to 5 (extremely important)

6.4 Added Benefits of Task-Based Policy Specification

Above, we described our approach to creating policy directly from a set of instantiated task templates which a battlefield commander could organize and prioritize into a mission plan. This approach was developed primarily as a means of reducing the workload required for a commander to create information management policy. In thinking about it under the AICE program, however, we have come to realize that it provides at least two additional benefits. These are described below.

6.4.1 Conditional Branching

One of the advantages that task- or mission plan-based approach to policy specification provides is that it greatly facilitates the incorporation of reasoning about various conditional branches in the battle plan. When a mission plan is created, it frequently contains such conditional branches, based on how the commander's forces should react if various outcomes are or are not achieved.

The status of such conditional branches clearly should have a profound effect on the commander's policies. For example, consider a unit that is held in reserve during a battle. There will be some condition (e.g., the commander's order, if the enemy reaches the bridge before we do, if they commit their reserves, etc.) upon which the reserves will shift from observing the battle and awaiting orders to actively engaging in it. Prior to that condition, their information needs will be for monitoring battle status, enemy position, and their commander's orders—perhaps with a level of priority lower than the troops actually engaged. Upon engagement, however, their need for overall battle status information probably drops, while their need for enemy position information rises.

For the commander or his staff to be able to assert these requirements in a 'raw' form whenever a conditional aspect of his battle plan becomes active is unrealistic. Far more plausible is the ability to simply assert that a conditional task (*with its associated information policy elements*) has become active whenever the commander orders it or the events in the world dictate it—and assume that the information needs and policy elements associated with that task are now active with an importance based on the commander's stated importance of the task.

Another, related advantage of linking policy elements to conditional tasks is that this allows the commander to plan for battlefield contingencies in a fashion that is familiar to him and his staff—by associated contingent tasks. Reasoning about when different priorities for information may be needed in the battle on the basis of, say, a temporal organization would, at best, obscure a chain of reasoning that is more natural for the command staff to make. While it may be possible for the command staff to say that C company will need weather information with high priority at 0800 hours, it will generally be more straightforward and explicit, not to mention less error prone, for them to say that C company will need weather information when it begins planning its route—a contingent task that is expected to start at 0800, but may start earlier, later or not at all.

Finally, the linking of information policy to tasks (some of which can be contingent) also facilitates reasoning about the probabilities with which information may actually be needed. If a probability of occurrence can be assigned to a contingent task along with a presumed importance for that task if it does occur, then stochastic optimization techniques can be used to factor the value of meeting some information requirements which, while not currently active, would be critically needed if their contingent task were to occur. This was essentially the forecasting work that Steve McBurnett and Metron proposed to do in their AICE IPM contract. Of course, such optimization techniques could be applied directly to information needs themselves, but this would be both more complicated and less natural for commanders. The aggregation of information needs by task, and then reasoning about current, planned and contingent tasks, is a natural simplification.

6.4.2 Precaching

When a task- or mission-plan based aggregation of information needs is laid out over time, either as a strict sequence or as a sequence with conditional branches, it offers yet another opportunity for improving human-machine system performance. In essence, because we know the likely information needs of tasks which have not yet occurred, we are in a position to pre-satisfy or "pre-cache" that information if excess bandwidth is available at any point before the task becomes active.

Such pre-caching would have to be dependent on the characteristics of the needed information and on the availability of unused communications resources. For example, it would make no sense to pre-cache enemy positions 24 hours before a battle task which needed enemy positions less than an hour old.

A more significant hurdle stems from the fact that there is a level of specificity increase between the policy element statements which commanders make in the current IPM implementation and the information needs statements which would need to be associated with specific tasks in order to permit pre-caching. A policy element statement is meant to define and match against a class or

set of specific information requests which requesters may issue; while pre-caching would probably require issuing only one or a few of these specific requests.

Taken together, the objections in the previous paragraphs imply that pre-caching will not be universally useful or effective. It is not a panacea. But the payoffs are potentially very high and the costs relatively low. If pre-cached information is sent only in spare bandwidth (that is, bandwidth that would otherwise go unused) and if we assume that pre-cached information simply resides on a local server until requested (and therefore, does not produce information overload for the human user), then *any* information request that is met by pre-cached information is a request that will not have to use network communications resources during a potentially critical period. Of course, making fewer requests means more bandwidth than would otherwise have been available to be used to meet other, non-pre-cached requests.

Section 7
Implementation

In this section, we offer some detail on the actual implementation of the concepts described in earlier sections and the environment used to test the implementation.

7.1 IPM Architecture

Figure 23 illustrates the basic architecture of the IPM implementation and defines the main components: the IPM Server itself, the information repository, a configuration file, an AIC server stub, the global clock server, and the IPM test driver. Each component is described here individually.

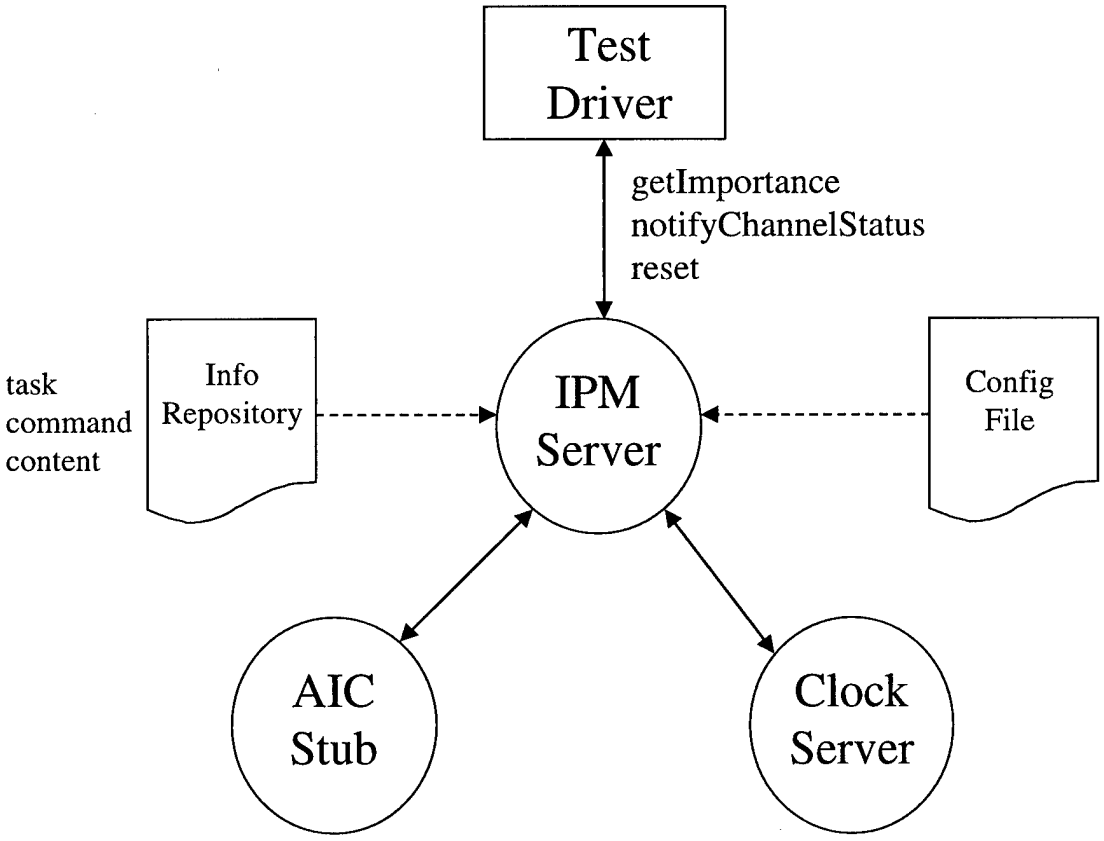

Figure 23. IPM Architecture

7.1.1 IPM Server

The IPM Server is the main component of the environment, implementing the core concepts of the IPM layer. Built in the programming language Java and using the object-broker standard CORBA as its main method of communication, the IPM Server has the following responsibilities:

- Upon initialization, the IPM Server reads policy and task statements from the repository and populates its internal data structures.
- Connects with various external servers as defined by the AICE requirements, namely the AIC server (representing the AIC layer) and the global clock server.
- Services *get-importance* and *notify-channel-status* requests as they come from the test driver during operation.
- Handles all *reset* requests from the test driver, which clears state and memory in the IPM Server and erases outstanding channel requests; The reset operation was created to allow for extensive testing of the IPM Server without re-initializing the entire AICE test environment.
- Maintains a log-file of all activities.

The IPM Server first performs initialization routines at start-up, and then waits for incoming requests from the test driver. Periodically, the test driver makes *get-importance* requests, which motivates the IPM Server to compare the request's parameters with the stated policies, before returning a calculated importance value. All *get-importance* requests include channel identifications that are stored by the IPM Server. The test driver may also request to update the state of the channel through the *notify-channel-status* operation. A requested channel may be of the state REQUESTED, ALLOCATED, TERMINATED, PREEMPTED, or REJECTED.

Future plans of the IPM Server include the ability to modify the stated policy and task descriptions dynamically in the information repository, and subsequently enhance the IPM Server with the capability to propagate the modifications automatically to active channel requests.

7.1.2 Test Driver

The test driver imitates the activity of the final complete AICE environment, in which requests are made of the IPM Server to identify importance of a channel requests based on stated policy. Access to the IPM Server is done remotely or locally via a CORBA interface, and results are stored to a log-file. Currently, the test driver can make three different requests:

- The *get-importance* operation requests the importance value of a given channel request requirements
- The *notify-channel-status* operation updates the state of an existing active channel
- The *reset* operation re-initializes the IPM Server; At least one *reset* request must be made before servicing the other operations

7.1.3 Information Repository

The information repository represents the actual information hierarchies that define policy. Currently implemented as a textual file and based on the eXchange Markup Language (XML) format standard, the information repository includes the following:

- The command hierarchy, describing the sources and destinations of potential channel requests

- The content hierarchy, representing the exchange characteristics of potential channel requests
- The task hierarchy, capturing the stated policies and policy elements

Future implementations will allow the information repository to be more interactive with the IPM Server, not just a static file-based entity.

7.1.4 Configuration File

The configuration file is a simple text file that defines host names and locations of the various servers to which the IPM Server must have access. Such servers presently include the AIC (stub) server and the global clock server. The configuration file allows the names and locations of the external servers to be parameterized, and not hard-wired into the IPM Server.

7.1.5 AIC Stub Server

The AIC Stub Server represents certain facets of the AIC layer, and serves as a temporary stub for testing purposes. The AICE requirements define a number of interface features that are not yet utilized by the entire AICE implementation, but are included in the IPM Server in preparation for further development. The AIC Stub Server is implemented in Java and uses CORBA to communicate with the IPM Server.

7.1.6 Global Clock Server

The global clock server maintains global time for the entire AICE environment. Implemented in Java and utilizing CORBA for access communications, the global clock server provides the means for all layers in the AICE architecture to synchronize to the same clock.

7.2 Building Information Hierarchies

As part of the IPM development, we have implemented a graphical tool that allows one to build pictorially the IPM policy descriptions and information hierarchies, and then automatically generate the required XML files for the information repository. In the future, these capabilities will be packaged into an interactive component that communicates directly with the IPM Server and provides dynamic modification of policies. As shown in Figure 24, the task hierarchy is described pictorially and allows one to define the policy elements under each task. From these descriptions, the user can automatically generate the required XML files for the IPM information repository. Individual policy elements have various user-defined properties, including owner, source, destination, exchange characteristics, and importance (Figure 25). The exchange characteristics and command hierarchies are also defined by the graphical tool, which also may be translated automatically to XML for the information repository (Figure 26).

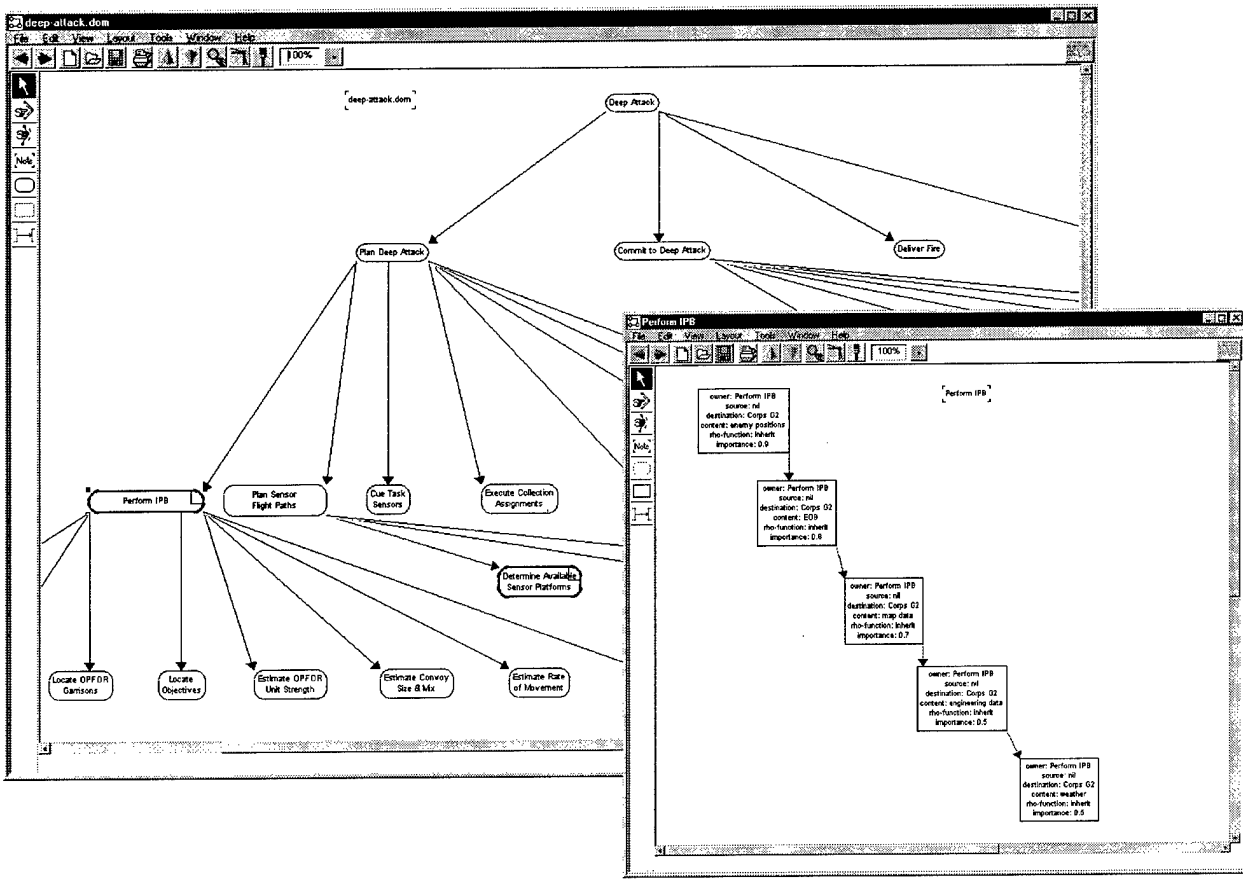

Figure 24. Task hierarchies are captured graphically.

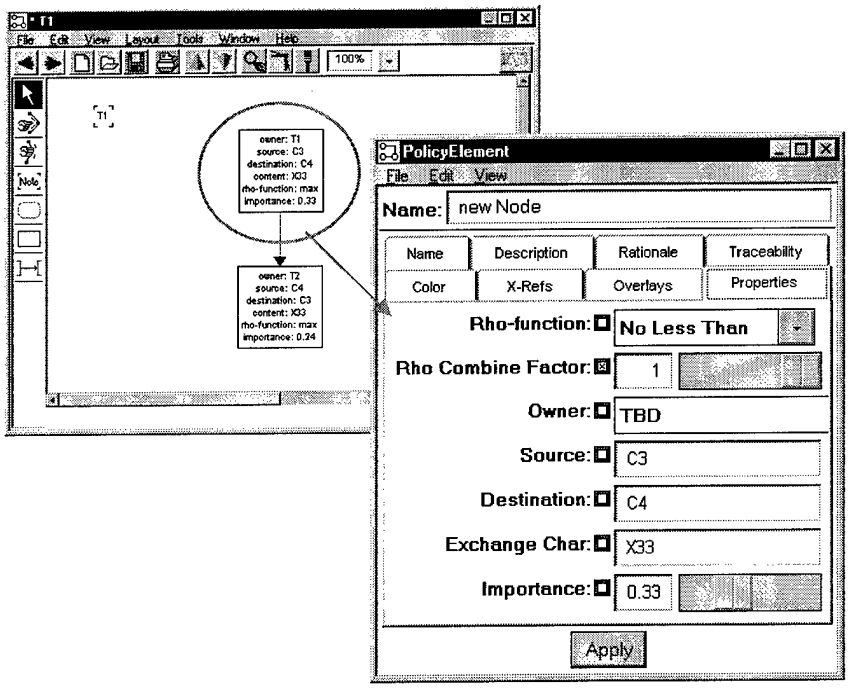

Figure 25. Policy Elements have modifiable properties.

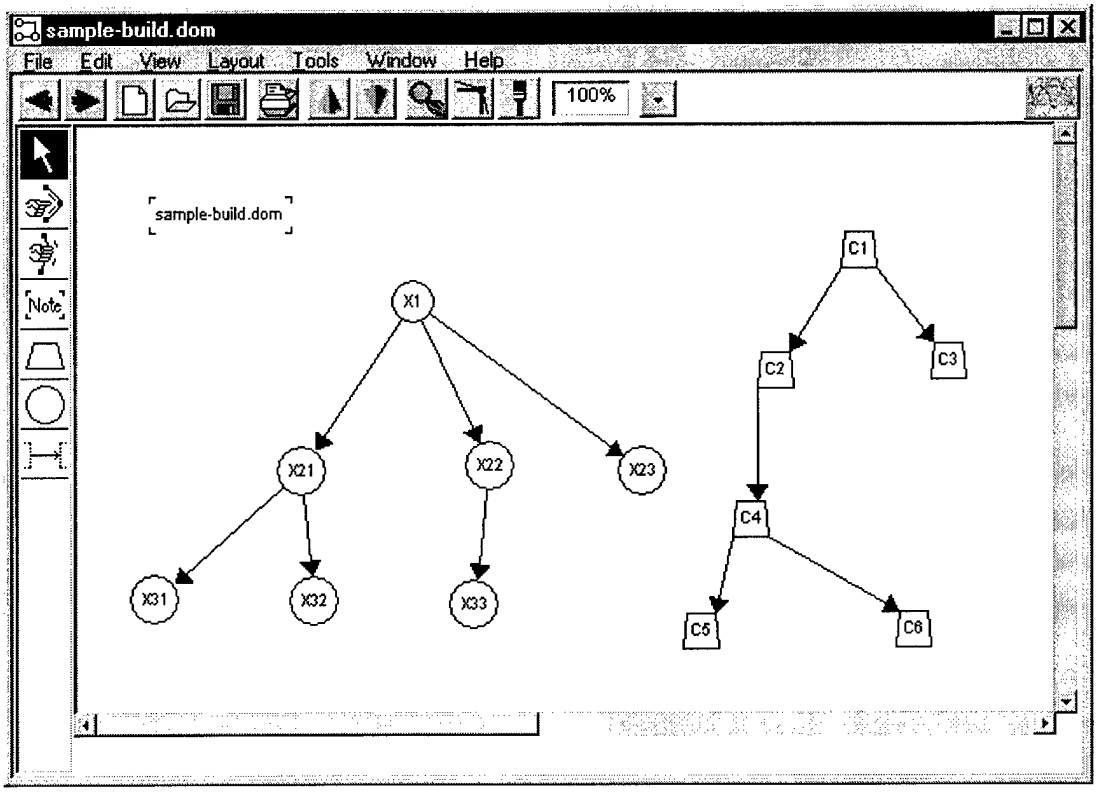

Figure 26. The Exchange Characteristics and Command Hierarchies

Section 8
References

1. The Defense Messaging Service. http://www.disa.mil/D2/dms/index2.html

2. McHargue, Maj. M. K., DMS X.500 Directory Services Overview. 1999 DMS Users Conference, Feb. 1999. http://www.disa.mil/D2/dms/uc99/brief/uc99dir1.zip.

3. Wahl, M., Howes, T., and Kille, S. Lightweight Directory Access Protocol (v3). IETF RFC 2251.

4. Howes, T. The String Representation of LDAP Search Filters. IETF RFC 2254.

5. Silvey, P. Agile Information Control Environment (AICE) Exchange Characterization Description and Specification, Draft v6.0, The MITRE Corporation, July 6, 1999.

6. Boy, G. (1999).

7. Universal Joint Task List, CJCSM 3500.04B, 01 November 1999. Available from DTIC, http://www.dtic.mil/doctrine/jel/cjcsd/cjcsm/m3500_4b.pdf.

8. Air Force Task List, Air Force Doctrine Document 1-1, 12 August 1998. Available from AFPUBS, http://afpubs.hq.af.mil/pubfiles/af/dd/afdd1-1/afdd1-1.pdf.

9. Anastasi, D., Hutton, R., Thorsden, M., Klein, G. & Serfaty, D. (1997). Cognitive Function Modeling for Capturing Complexity in System Design. In Proceedings of the IEEE conference on Systems, Man and Cybernetics, October. 221-226.

10. Miller, C. (1999). Bridging the Information Transfer Gap: Measuring Goodness of Information Fit. Journal of Visual Languages and Computing, 10. 523-558.

11. Miller, C. & Hannen, M.D. (1999). The Rotorcraft Pilot's Associate: Design and Evaluation of an Intelligent User Interface for Cockpit Information Management. Knowledge-Based Systems, 12. 443-456.

12. Miller C., and Funk, H. (1996). Knowledge Requirements for Information Management; A Rotorcraft Pilot's Associate Example. In M. Mouloua, J. Koonce (eds.). Human-Automation Interaction: Research and Practice, Lawrence Erlbaum, Hillsdale, NJ, 1996.

13. Miller C., & Goldman, R. (1997). 'Tasking' Interfaces: Associates that know who's the boss. Proceedings of the 4th USAF/RAF/GAF Conference on Human/Electronic Crewmembers. Kreuth, Germany.

14. Miller, C.A., Pelican, M. & Goldman, R. (2000). Tasking Interfaces to Keep the Operator in Control. Proceedings of the 5th Annual Symposium on Human Interaction with Complex Systems. April 30-May 2, Urbana, Ill.

15. Wolf, S., Klein, G., & Serfaty, D. (1996). A Model of Information Management, Technical Memorandum, July 1999.

16. Chi, M.T.H., Feltovich, P.J., and Glaser, R. (1981). Categorization and representation of physics problems by experts and novices. Cognitive Science. 5, 121-152.

17. Pennington, Nancy. (1987). Comprehension strategies in programming. In Olson, Gary M.; Sheppard, Sylvia; and Soloway, Elliot (Eds.), Empirical Studies of Programmers: Second Workshop. Norwood, NJ: Ablex Publishing Corp.